

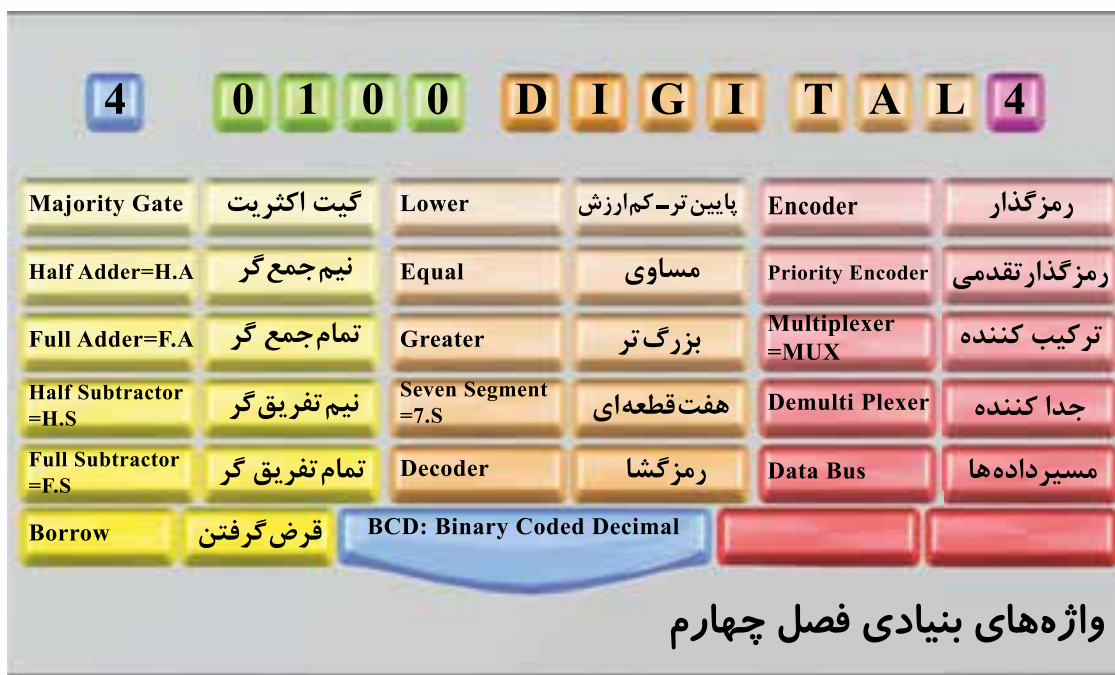
چند مدار ترکیبی کاربردی

هدف کلی: بررسی و طراحی مدارهای ترکیبی که وضعیت خروجی‌های آن در هر لحظه منحصرأ به وضعیت ورودی‌های آن، در همان لحظه بستگی دارد و مدارهای ترکیبی با کاربردهای ویژه که به علت مصرف عام به صورت تراشه‌های تجارتي عرضه می‌شوند.

کل زمان اختصاص داده شده به فصل: ۲۴ ساعت آموزشی

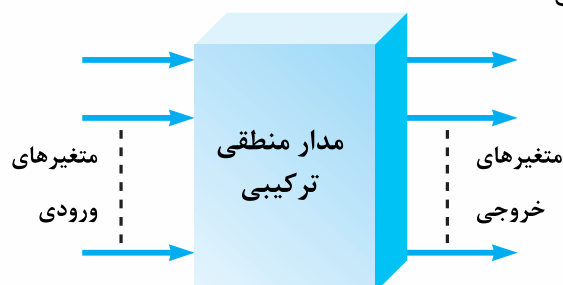
هدف‌های رفتاری: در پایان این فصل از فراگیرنده انتظار می‌رود که:

- ۱- مدارهای ترکیبی را تعریف کند.
- ۲- روش طراحی مدارهای ترکیبی را توضیح دهد.
- ۳- چند نمونه مدار ترکیبی را طراحی کند.
- ۴- مدارهای ترکیبی با کاربردهای ویژه را تعریف کند.
- ۵- مدار نیم جمع‌کننده (H.A) را تحلیل کند.
- ۶- مدار تمام جمع‌کننده (F.A) را تحلیل کند.
- ۷- مدار تفريق‌کننده ناقص (H.S) را تحلیل کند.
- ۸- مدار تفريق‌کننده کامل (F.S) را تحلیل کند.
- ۹- انواع کدها را تعریف کند.
- ۱۰- انواع کد را با کد BCD مقایسه کند.
- ۱۱- مبدل هگزادسی‌مال به 7.S (سون سگمنت) را با VLSI توضیح دهد.
- ۱۲- مبدل BCD به 7.S را با رسم جدول شرح دهد.
- ۱۳- مدار مقایسه‌کننده یک بیتی را تحلیل کند.
- ۱۴- مدار رمزگشا (Decoder) را تحلیل کند.
- ۱۵- توابع منطقی را با رمزگشا اجرا کند.
- ۱۶- مدار رمزگذار (Encoder) را تحلیل کند.
- ۱۷- مدار متمرکزکننده (مالتی پلکسر) را تحلیل کند.
- ۱۸- نحوه افزایش ظرفیت متمرکزکننده‌ها را تحلیل کند.
- ۱۹- انتقال اطلاعات به کمک متمرکزکننده‌ها را تحلیل کند.
- ۲۰- توابع منطقی را توسط مالتی پلکسر (متمرکزکننده‌ها) اجرا کند.
- ۲۱- نمایشگرهای الکترونیکی (7.S و LED) را توضیح دهد.
- ۲۲- 7.S کاتد مشترک، آند مشترک و آی‌سی‌های مربوطه را تشریح کند.
- ۲۳- چیپ‌ها (Chips) را از نظر گروه LSI , MSI , SSI و VLSI توضیح دهد.
- ۲۴- با استفاده از data book آی‌سی‌های مدارهای ترکیبی ویژه را شناسایی کند.
- ۲۵- با کمک نرم‌افزار مولتی سیم مدارهای ترکیبی را شبیه‌سازی کند.
- ۲۶- به سؤالات الگوی پرسش پاسخ دهد.
- ۲۷- کلیه هدف‌های رفتاری در حیطه عاطفی که در فصل اول آمده است را باید در این فصل مورد توجه قرار دهد.



اندازه کافی صبر کنیم تا روی خروجی‌ها اثر بگذارند، در این شرایط مقادیر خروجی‌ها تحت تأثیر آخرین مجموعه مقادیر ورودی‌ها قرار می‌گیرند و مقدار آنها را مشخص می‌کند.

به آنچه که از مدارهای ترکیبی بیان شد «رفتار مدار» گفته می‌شود. می‌توانیم مدارهای ترکیبی را بر اساس ساختار مدار نیز تعریف کنیم. به عبارت ساده، مدارهای ترکیبی مدارهایی بدون فیدبک (پس‌خورد) و بدون عنصر حافظه هستند. یک مدار ترکیبی شامل متغیرهای ورودی، دروازه‌های منطقی و متغیرهای خروجی است. شکل ۴-۱ بلوک دیاگرام یک مدار ترکیبی را نشان می‌دهد.



شکل ۴-۱- بلوک دیاگرام یک مدار ترکیبی

پیش‌گفتار

در فصل سوم به بررسی جبر بول پرداختیم و توابع جبری آن را به صورت ساده شده نشان دادیم و اشاره‌ای به مدارهای ترکیبی داشتیم، در این فصل به بررسی مدارهای منطقی ترکیبی با کاربرد ویژه می‌پردازیم. در طراحی و ساخت مدارهای منطقی باید تا حد امکان از ساده‌ترین و کم‌ترین قطعات استفاده شود. به عبارت دیگر، در طراحی مدارهای منطقی باید ابتدا مدارها را با روش‌های مختلف خلاصه و ساده کنید، تا به یک مدار بهینه قابل قبول با حداقل قطعات برسید. در این فصل، به بررسی مدارهای ترکیبی که کاربری عمومی دارند می‌پردازیم و با زبان ساده آنها را تشریح می‌کنیم.

۴-۱- مدارهای ترکیبی

مدارهای ترکیبی، مدارهایی هستند که خروجی‌های آنها به طور هم‌زمان به ورودی‌های آنها بستگی دارد. به عبارت دیگر اگر ورودی‌هایی را به شبکه اعمال کنیم و به

۴-۱-۱- روش طراحی مدارهای ترکیبی: در فصل

سوم پیاده‌سازی روابط منطقی را با استفاده از گیت‌ها بیان کردیم. در این فصل به طراحی مدارهای ترکیبی می‌پردازیم.

طراحی مدارهای ترکیبی با تعریف یک مسئله شروع می‌شود و با دیاگرام «منطقی مدار» یا «مجموعه‌ای از توابع بول» که با استفاده از آنها می‌توان به سادگی دیاگرام منطقی را به‌دست آورد پایان می‌یابد. مراحل زیر روند طراحی مدارهای ترکیبی را نشان می‌دهد.

۱- تعریف دقیق مسئله.

۲- تعیین تعداد ورودی‌ها و خروجی‌های لازم.

۳- تشکیل جدول درستی مدار که ارتباط بین ورودی‌ها و خروجی‌ها را برقرار کند.

۴- نوشتن تابع منطقی.

۵- ساده‌سازی تابع منطقی بولی به‌دست آمده برای هر یک از خروجی‌های مدار.

۶- رسم مدار منطقی با حداقل گیت یا با گیت‌های خواسته شده.



نکته: در طراحی مدار سعی می‌کنیم تعداد دروازه‌های منطقی و تعداد ورودی‌های آن حداقل باشد.

جدول درستی یک مدار ترکیبی از ستون‌های ورودی و ستون‌های خروجی تشکیل می‌شود.

بر اساس صورت مسئله، با استفاده از 2^n حالت ورودی حالت‌های خروجی را به‌دست می‌آوریم. مشخصات مسئله ممکن است به گونه‌ای باشد که بعضی از ترکیب‌های ورودی هرگز به وجود نیایند. که این حالت‌ها را «حالت‌های بی‌اهمیت» (don't care) می‌گویند.

۴-۱-۲- طراحی چند نمونه مدار ترکیبی: در

فصل سوم، طراحی مدارهای مختلف ترکیبی را فرا

گرفتید، برای یادآوری و تأکید بیشتر در این فصل به شرح چند مثال دیگر می‌پردازیم.

مثال ۴-۱: مداری با سه ورودی A، B و C طراحی کنید، که اگر ورودی B یک باشد خروجی یک شود.

حل:

الف) رسم جدول صحت: ابتدا جدول صحت را با سه متغیر ورودی و یک خروجی رسم می‌کنیم. با توجه به صورت مسئله در جدول صحت، در شرایطی که ورودی B یک است، خروجی را یک می‌نویسیم.

جدول ۴-۱- جدول صحت مربوط به مثال ۴-۱

شماره سطر	A	B	C	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

ب) نوشتن تابع منطقی مدار: تابع منطقی را با توجه به حالت‌هایی که خروجی یک است می‌نویسیم برای نوشتن تابع منطقی از حاصل جمع حاصل ضرب‌ها (مین‌ترم) استفاده می‌کنیم.

$$F = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$

←→ سطر ۲
 ←→ سطر ۳
 ←→ سطر ۶
 ←→ سطر ۷

ج) ساده کردن تابع منطقی: از سطر ۲ و ۶ عبارت $\overline{B}C$ و از سطر ۳ و ۷ عبارت $\overline{A}C$ را فاکتورگیری می‌کنیم.

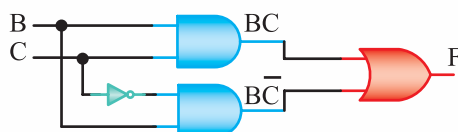
$$F = \overline{B}C(\overline{A} + A) + \overline{A}C(\overline{B} + B)$$

می‌دانیم هر متغیر که با مکمل خودش جمع شود حاصل آن یک است. هم‌چنین اگر «یک» منطقی در عبارتی ضرب شود، حاصل همان عبارت خواهد بود.

بنابراین عبارت خروجی به صورت زیر در می‌آید.

$$F = \overline{B}C + BC$$

(د) **طراحی مدار:** با توجه به عبارت منطقی به دست آمده برای تابع F ، مدار شکل ۴-۲ را طراحی می‌کنیم.



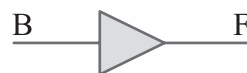
شکل ۴-۲- مدار مربوط به مثال ۴-۱

کمی به عبارت خروجی F توجه کنید، در می‌یابید که می‌توانید تابع F را ساده‌تر کنید. از متغیر B فاکتورگیری می‌کنیم، حاصل به صورت زیر در می‌آید.

$$F = B(\overline{C} + C) = B$$

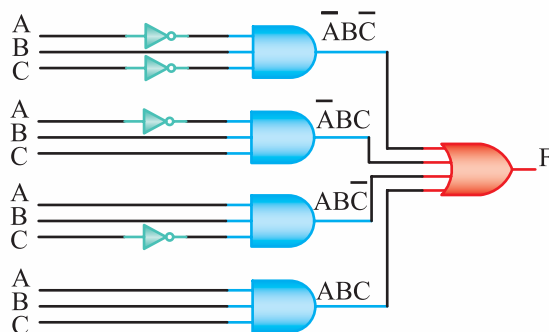
در نتیجه وقتی خروجی یک است که ورودی B یک باشد. به عبارت دیگر، سایر متغیرهای ورودی تأثیری در خروجی ندارند و می‌توانند حذف شوند.

مدار شکل ۴-۳ مدار ساده شده عبارت خروجی F است.



شکل ۴-۳- مدار ساده شده مثال ۴-۱

اگر از قوانین مربوط به ساده کردن عبارت‌های منطقی استفاده نمی‌کردیم باید مدار ساده مثال ۴-۱ را به صورت مدار پیچیده شکل ۴-۴ طراحی کنیم.



شکل ۴-۴- مدار مثال ۴-۱ قبل از ساده شدن

با مقایسه مدارهای شکل ۴-۳ و ۴-۴ در می‌یابیم که

تعداد گیت‌های استفاده شده در مدار ساده شده فقط یک گیت بافر و در مدار ساده نشده ۹ گیت مختلف و پیچیده است. لذا از این تفاوت نمی‌توان به راحتی عبور کرد.

مثال ۴-۲: مداری طراحی کنید که یک قفل را با ۳ کلید دو وضعیتی A ، B و C کنترل کند. رمز قفل در حالتی باز می‌شود که فقط یک کلید بسته باشد، (بسته بودن به مفهوم «یک» منطقی است).

حل:

(الف) **ترسیم جدول صحت:** ابتدا با توجه به تعداد متغیرها (۳ متغیر) و تعداد حالت‌ها ($2^3=8$) جدول صحت را مطابق جدول ۴-۲ رسم می‌کنیم. در این جدول ستون ۱ شماره سطر و ستون ۲ حالت‌های ورودی‌ها را نشان می‌دهد.

جدول ۴-۲- جدول صحت مربوط به مثال ۴-۲

شماره سطر	A	B	C	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

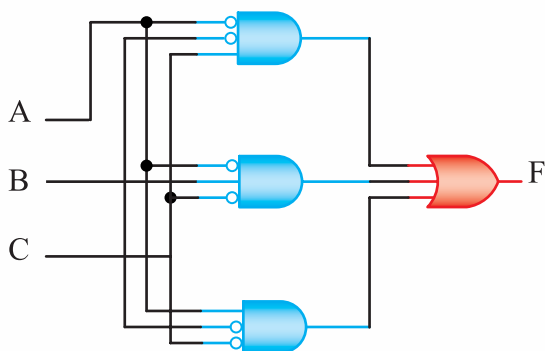
ستون ۴ همان خروجی مدار یا F است که بر اساس تعریف کارمدار، آن را تکمیل می‌کنیم. بنابر صورت مسئله، اگر فقط یکی از ورودی‌ها، یک و بقیه صفر باشند، ستون خروجی یعنی F یک خواهد شد.

(ب) **نوشتن تابع منطقی مدار:** حال باید تابع منطقی را برای حالت‌هایی که مقدار خروجی یک است، بنویسیم. برای نوشتن تابع منطقی می‌توانیم از حاصل جمع حاصل ضرب‌ها (مین ترم) استفاده کنیم.

$$F = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C}$$

← سطر ۱
← سطر ۲
← سطر ۴

با استفاده از نماد جدید، مدار شکل ۴-۵ را می‌توان به صورت ساده‌تر (شکل ۴-۸) درآورد.



شکل ۴-۸- مدار ساده‌شده مثال ۴-۲

مثال ۴-۳: یک ساختمان دارای سه طبقه است. برق سه فاز وارد ساختمان می‌شود و هر طبقه را با یک فاز تغذیه می‌کند. شرایط کار در ساختمان به گونه‌ای است که اگر برق دو طبقه قطع شود (دو فاز قطع شود)، باید حتماً طبقه‌ای که برق دارد نیز قطع شود. مدار طراحی کنید که توانایی وصل بودن حداقل دو فاز را نشان دهد.

حل:

مرحله اول: برای حل این مسئله ابتدا باید تعداد ورودی‌ها را مشخص کنیم. چون مدار سه فاز است، در نتیجه سه ورودی داریم.

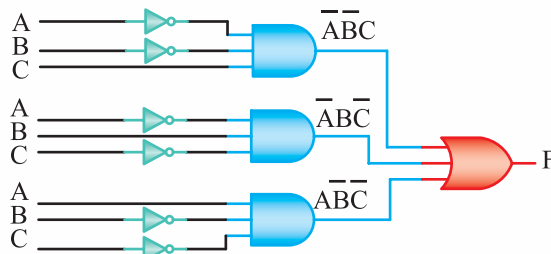
مرحله دوم: رسم جدول صحت و تعیین حالت‌هایی است که خروجی جواب یک دارد. جدول ۴-۳ جدول صحت این مدار را نشان می‌دهد.

جدول ۴-۳- جدول صحت مثال ۴-۳

شماره سطر	A	B	C	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

ج) ساده‌کردن تابع منطقی: تابع منطقی به دست آمده ساده نمی‌شود.

د) طراحی مدار: برای طراحی مدار ابتدا از تابع منطقی به دست آمده (F) کمک می‌گیریم، سپس با استفاده از دروازه‌های منطقی AND، OR و NOT مدار را طراحی می‌کنیم. شکل ۴-۵ مدار مربوط به مثال ۲ را نشان می‌دهد.



شکل ۴-۵- مدار مثال ۴-۲

همان‌طور که در شکل ۴-۵ مشاهده می‌کنید بعضی از ورودی‌ها برای اتصال به گیت بعدی، ابتدا وارد گیت منطقی NOT می‌شوند، تا به صورت متمم درآیند. برای هر یک از گیت‌های NOT، یک نماد مداری مشابه شکل ۴-۶ رسم شده است.



شکل ۴-۶- دروازه منطقی NOT

برای ساده‌تر شدن مدارهای ترسیمی، در رسم مدارها معمولاً گیت NOT را با یک دایره تو خالی در ورودی گیت مورد نظر (مثلاً گیت AND) نشان می‌دهند، در شکل ۴-۷ این اتصال را مشاهده می‌کنید.



علامت دایره تو خالی به جای گیت NOT استفاده شده است.

شکل ۴-۷- نماد گیت NOT به صورت دایره تو خالی

که قدرت انجام عملیات ریاضی جمع، تفریق، ضرب، تقسیم و... را داشته باشند. می‌دانیم کلیه عملیات ریاضی بر اساس جمع و تفریق صورت می‌گیرد، زیرا اجرای سایر عملیات به کمک جمع و تفریق امکان‌پذیر است. مثلاً برای اجرای ضرب 3×9 می‌توانیم ۹ را سه بار متوالی با خودش جمع کنیم، یعنی:

$$3 \times 9 = 9 + 9 + 9$$

یا برای انجام تقسیم $15 \div 3$ ، می‌توانیم ۳ را پنج بار متوالی از ۱۵ کم کنیم، و آخرین مرحله که باقی‌مانده مساوی صفر می‌شود، جواب تقسیم است. یعنی:

- | | |
|---|---------------|
| ① | $15 - 3 = 12$ |
| ② | $12 - 3 = 9$ |
| ③ | $9 - 3 = 6$ |
| ④ | $6 - 3 = 3$ |
| ⑤ | $3 - 3 = 0$ |

۵ جواب تقسیم یا خارج قسمت است. لذا به دلیل این که جمع و تفریق اساس عملیات ریاضی را تشکیل می‌دهد. در این مبحث به بررسی جمع‌کننده و تفریق‌کننده می‌پردازیم.

۴-۲-۱- جمع‌کننده ناقص H.A (Half Adder):

جمع دورقم دودویی را می‌توان با مداری به نام جمع‌کننده ناقص یا به اختصار H.A انجام دهیم. مدار H.A مداری است که دو ورودی (A و B) و دو خروجی (S و C) دارد. S رقم اول حاصل جمع SUM و C رقم نقلی یا Carry را مشخص می‌کند.

جدول صحت جمع دوییت A و B را در جدول ۴-۴ مشاهده می‌کنید.

جدول ۴-۴- جدول صحت نیم‌جمع‌کننده

A	B	C	S	
0	0	0	0	$0 + 0 = 0$
0	1	0	1	$0 + 1 = 1$
1	0	0	1	$1 + 0 = 1$
1	1	1	0	$1 + 1 = 10$

مرحله سوم: عبارت‌های منطقی مربوط به خروجی یک را می‌نویسیم.

$$F = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

← سطر ۳
← سطر ۵
← سطر ۶
← سطر ۷

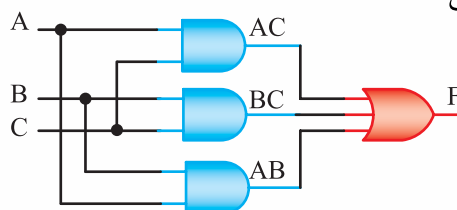
مرحله چهارم: عبارت خروجی را ساده می‌کنیم.

$$F = BC(\overline{A} + A) + AC(\overline{B} + B) + AB(\overline{C} + C)$$

← سطر ۳ و ۷
← سطر ۵ و ۷
← سطر ۶ و ۷

مرحله پنجم: طراحی و رسم مدار مربوط به

خروجی



شکل ۹-۴- مدار مثال ۳-۴

تمرین کلاسی ۴-۱: گیت اکثریت (Majority gate)

یک مدار دیجیتالی است که اگر اکثر ورودی‌های آن (بیش از ۵۰ درصد) یک باشد، خروجی آن نیز یک است. همچنین اگر اکثر ورودی‌های مدار صفر شود، خروجی نیز صفر خواهد بود. حال اگر مدار چهار ورودی داشته باشد، تابع بولی مناسب را بنویسید و مدار آن را طراحی کنید.

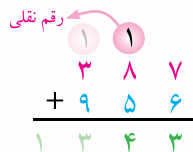
۴-۲- مدارهای ترکیبی با کاربردهای ویژه

در این بخش، بعضی از مدارهای ترکیبی با کاربردهای ویژه، که به علت مصرف عام به صورت تراشه‌ها (IC) تجارتي عرضه می‌شوند را معرفی می‌کنیم. ابتدا در هر مورد به بیان اصول کار مدار با تکیه بر مفاهیم اساسی جبر بول می‌پردازیم، سپس بحث را با معرفی تراشه‌های تجارتي به پایان می‌رسانیم.

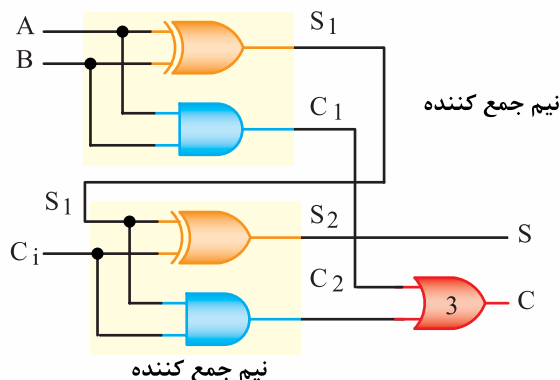
ماشین‌های محاسبه‌گر دیجیتالی باید مدارهایی باشند

جمع‌کننده کامل یا تمام‌جمع‌گر مداری است که ۳ خط ورودی (A، B و C_i) و دو خط خروجی (S و C) را دارد. به این ترتیب مدار جمع‌کننده کامل می‌تواند دو رقم دودویی و یک Carry که از مرحله قبل حاصل شده است را با هم جمع کند. با بیان مثالی در جمع اعداد ده‌دهی درک موضوع را ساده‌تر می‌کنیم.

در جمع دو عدد ۹۵۶ و ۳۸۷ وقتی می‌خواهیم دو رقم دهگان ۵ و ۸ را جمع کنیم ابتدا باید حاصل جمع دو رقم یکان را به دست آوریم از حاصل جمع دو رقم یکان ۶ و ۷ عدد ۱۳ حاصل می‌شود که ۳ را در ستون یکان می‌نویسیم و رقم یک را به ستون دهگان انتقال می‌دهیم که اصطلاحاً ده بر یک می‌گوییم.



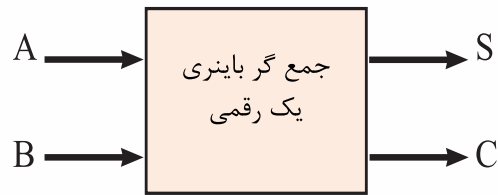
عدد یک، رقم نقلی است که در جمع ۸ و ۵ تأثیر می‌گذارد در سیستم باینری نیز به همین ترتیب رقم نقلی وجود دارد و روی عمل جمع اثر می‌گذارد. به کمک دو نیم‌جمع‌کننده می‌توان یک تمام‌جمع‌گر یا یک جمع‌گر یک‌بیتی کامل مطابق شکل ۴-۱۲ ساخت.



شکل ۴-۱۲ مدار تمام جمع‌گر یک بیتی

به‌طور مثال برای جمع دو عدد باینری چهار بیتی ۱۰۱۱ و ۱۱۰۱ خواهیم داشت:

بلوک دیاگرام مداری که بتواند دو رقم باینری را با هم جمع کند، در شکل ۴-۱۰ نشان داده‌ایم.



شکل ۴-۱۰ بلوک دیاگرام جمع‌کننده ناقص

با توجه به جدول ۴-۴ به این نتیجه می‌رسیم که اگر فقط دو بیت با هم جمع شوند، رقم نقلی از قبل وجود ندارد و جمع‌کننده ناقص (نیم‌جمع‌کننده) می‌تواند این عمل جمع را انجام دهد.

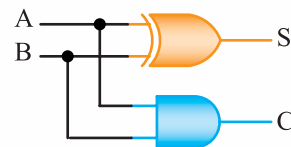
مطابق جدول صحت مدار، تابع S شامل دو جمله $\overline{A}B$ و $A\overline{B}$ است، یعنی:

$$S = \overline{A}B + A\overline{B} = A \oplus B$$

از طرفی تابع C فقط شامل یک جمله AB است یعنی:

$$C = AB$$

با استفاده از مقادیر S و C به کمک یک دروازه XOR و یک دروازه AND می‌توانیم مداری مطابق شکل ۴-۱۱ را طراحی کنیم.



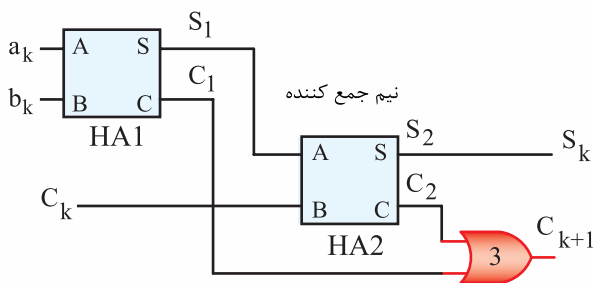
شکل ۴-۱۱ مدار نیم جمع‌گر باینری یک رقمی

در این جمع‌گر، ورودی رقم نقلی پیش‌بینی نشده است. به همین دلیل، به آن جمع‌گر ناقص یا نیم‌جمع‌گر می‌گویند.

۴-۲-۲- جمع‌کننده کامل F.A (Full Adder):

برای انجام عملیات جمع اعداد دودویی نیاز به مداری داریم که بتواند ۳ رقم یک بیتی باینری را با هم جمع کند. چنین مداری را جمع‌کننده کامل می‌گویند و با F.A نشان می‌دهند.

مستلزم آن است که هر دو نیم جمع کننده رقم نقلی یک ایجاد کنند، و این تنها در صورتی تحقق می‌یابد که هر دو ورودی نیم جمع کننده یک باشند.



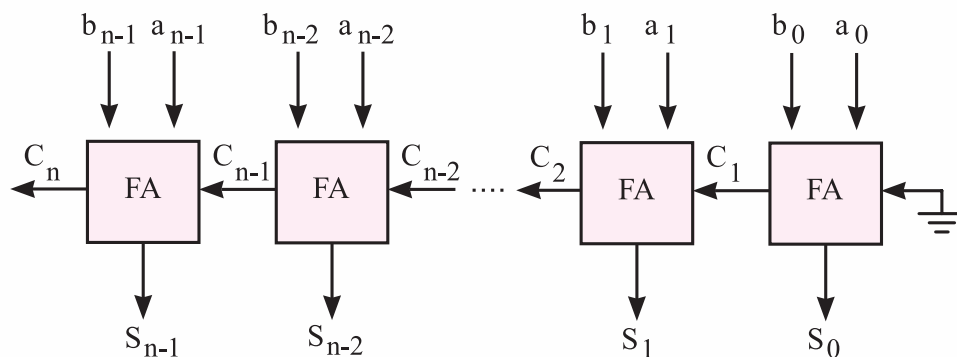
شکل ۴-۱۳: بلوک دیاگرام تمام جمع گر

حال اگر هر دو ورودی a_k و b_k برابر یک باشند، خروجی $S_1 = 0$ و $C_1 = 1$ می‌شود. چون خروجی $S_1 = 0$ است، خروجی $C_2 = 0$ می‌شود، (آیا می‌دانید چرا؟) اگر ورودی‌های a_k و b_k هم‌زمان یک نشوند، خروجی $C_1 = 0$ می‌شود؛ یعنی، هرگز خروجی‌های C_1 و C_2 هم‌زمان یک نمی‌شوند.

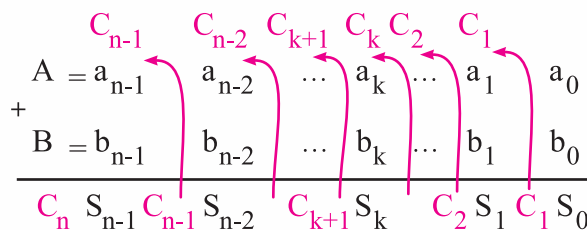
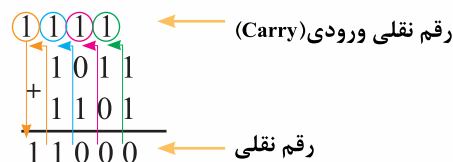


تمرین کلاسی ۲-۴: جدول صحت یک تمام جمع گر یک بیتی را بنویسید و به کمک آن توابع S_k و C_{k+1} را بر حسب ورودی‌های a_k ، b_k و C_k به دست آورید.

برای جمع کردن دو عدد n رقمی، باید n طبقه تمام جمع گر را مطابق شکل ۴-۱۴ پشت سر هم ببندیم.



شکل ۴-۱۴: بلوک دیاگرام یک جمع گر n بیتی



یعنی، تمام جمع‌گری که در موقعیت مکانی K قرار می‌گیرد، باید سه بیت C_k ، a_k و b_k را با یک دیگر جمع کند.

$$C_k + a_k + b_k$$

رابطه بالا را می‌توانیم به صورت زیر بنویسیم.

$$C_k + a_k + b_k = C_k + (a_k + b_k)$$

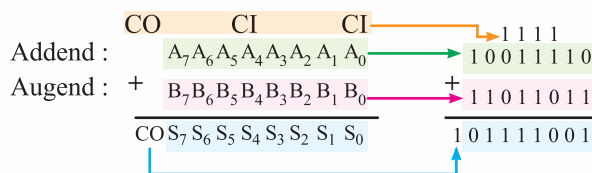
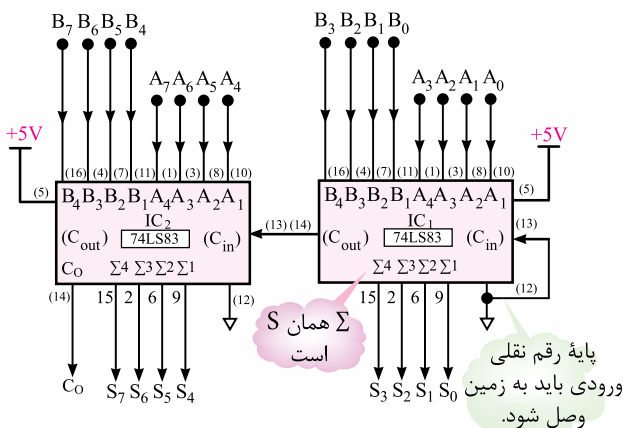
یعنی، ابتدا ارقام a_k و b_k را با هم جمع کنیم و سپس حاصل جمع آن‌ها را با C_k جمع می‌کنیم. در طراحی مدار نیز باید به همین ترتیب عمل کنیم. در شکل ۴-۱۳ مدارهای نیم‌جمع کننده شکل ۴-۱۲ به صورت بلوک دیاگرام نشان داده شده است در این مدار ابتدا نیم‌جمع کننده HA_1 ارقام a_k و b_k را با هم جمع می‌کند و سپس نیم‌جمع کننده HA_2 حاصل جمع این دو رقم را با C_k به دست می‌آورد.

توجه داشته باشید که هرگز ورودی‌های دروازه OR (گیت شماره ۳) هم‌زمان یک نمی‌شود، زیرا این امر

به زمین اتصال داده می‌شود. دو عدد هشت بیتی $B_7 B_6 B_5 B_4 B_3 B_2 B_1 B_0$ و $A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0$ ابتدا در آی‌سی IC_1 (سمت راستی) ارقام $A_3 A_2 A_1 A_0$ با $B_3 B_2 B_1 B_0$ جمع می‌شوند و اگر رقم نقلی ایجاد شود به پایه ۱۴ آی‌سی سمت چپ انتقال می‌یابد. سپس ارقام $A_5 A_4$ و $B_5 B_4$ با $A_7 A_6$ و $B_7 B_6$ جمع شده و در صورتی که رقم نقلی ایجاد شود در پایه ۱۴ آی‌سی دوم ظاهر می‌شود. حاصل جمع چهار بیت کم ارزش‌تر در پایه‌های ۹ و ۶ و ۲ و ۱۵ آی‌سی سمت راست و حاصل جمع چهار بیت با ارزش بالاتر را در پایه‌های ۹ و ۶ و ۲ و ۱۵ آی‌سی سمت چپ مشاهده می‌کنید.

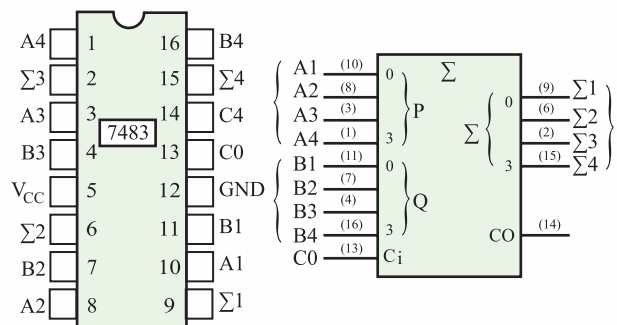


جهت هنجاریان علاقه‌مند: با استفاده از نرم‌افزار مولتی سیم یا پروتئوس دو عدد آی‌سی ۷۴۸۳ شکل ۱۶-۴ را ببینید و با تغییر ورودی‌ها به حالت صفر و یک نتیجه را در خروجی‌های مدار مشاهده کنید.



شکل ۱۶-۴- چگونگی انجام دادن عمل جمع دو عدد هشت بیتی به کمک دو جمع‌گر ۷۴LS۸۳

مطابق این الگو، خروجی‌های طبقه اول- یعنی S_0 و C_1 بلافاصله پایدار می‌شوند، ولی خروجی‌های طبقه دوم، ابتدا حاصل جمع a_1+b_1 را نشان می‌دهند و پس از رسیدن C_1 به حاصل جمع پایدار خود می‌رسند. سایر طبقات نیز تا رسیدن رقم نقلی طبقات ماقبل به حالت پایدار نمی‌رسند. این نوع جمع‌گر را که در آن رقم نقلی به‌صورت موجی از طبقه اول ایجاد می‌شود و به‌تدریج طبقات دیگر را تحت تأثیر قرار می‌دهد، جمع‌گر موازی با رقم نقلی موجی (Parallel Ripple Carry Adder) می‌نامند. تأخیر انتشار رقم نقلی در مواردی که تعداد ارقام دو عدد زیاد باشد موجب کندی عمل جمع می‌شود. در جمع‌گرهای با سرعت زیاد با به‌کار بردن شیوه‌های خاصی رقم نقلی هر طبقه را پیش‌بینی و مستقیماً ایجاد می‌کنند. جمع‌گرهای باینری چهار بیتی با رقم نقلی موجی به‌صورت آی‌سی عرضه می‌شود. نمونه‌هایی از این آی‌سی‌ها به‌صورت تجارتي ۱۶ پایه با شماره‌های ۷۴LS۸۳، ۷۴۸۳A و ۷۴HC۸۳ در بازار وجود دارند. در شکل ۱۵-۴ مشخصات تراشه ۷۴۸۳A را مشاهده می‌کنید.



شکل ۱۵-۴- مشخصات جمع‌گر چهار بیتی ۷۴۸۳A

برای جمع کردن اعداد بزرگ‌تر از چهار رقم می‌توان از چند تراشه تجارتي استفاده کرد.

در شکل ۱۶-۴، چگونگی انجام دادن عمل جمع دو عدد هشت بیتی را با استفاده از دو جمع‌گر چهار بیتی ۷۴LS۸۳ مشاهده می‌کنید. در تراشه سمت راست چون رقم نقلی از مرحله اولی وجود ندارد پایه ۱۳ این آی‌سی

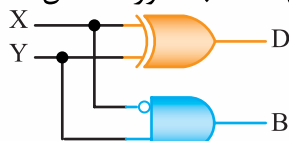
بر اساس جدول تفریق کننده ناقص عبارت مربوط به خروجی‌ها را می‌نویسیم.

$$D = \overline{xy} + x\overline{y} = x \oplus y$$

$$B = \overline{xy}$$

جالب توجه است که مقادیر خروجی D در اینجا با مقادیر خروجی در نیم جمع کننده یکسان است.

با استفاده از مقادیر به دست آمده در تابع خروجی، مدار نیم تفریق کننده به صورت شکل ۴-۱۷ در می‌آید.

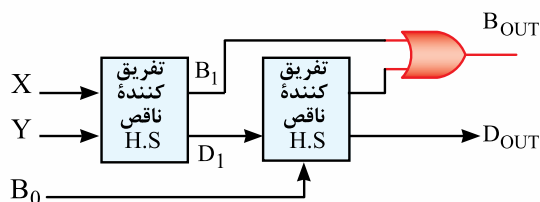


شکل ۴-۱۷ مدار نیم تفریق کننده

۴-۲-۴ تفریق کننده کامل F.S

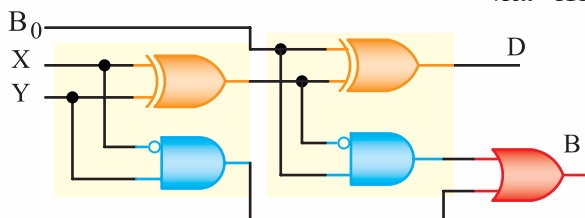
(Full Subtractor): می‌توان مطابق روشی که برای مدارهای «تمام جمع کننده» ذکر شده مدار تمام تفریق کننده یا تفریق کننده کامل را نیز با دو تفریق کننده ناقص و یک گیت OR ساخت.

شکل ۴-۱۸ بلوک دیاگرام تفریق کننده کامل را نشان می‌دهد.



شکل ۴-۱۸ بلوک دیاگرام تفریق کننده کامل

در شکل ۴-۱۸، رقم قرض گرفته شده از مرحله قبل است. مشابه آن چه که برای مدارهای تمام جمع کننده گفته شد.



شکل ۴-۱۹ مدار تفریق کننده کامل



جهت هنجاریان علاقه‌مند: با استفاده از نرم‌افزار

مولتی سیم مدار تمام جمع گر ۴ بیتی و تمام جمع گر هشت بیتی را ببندید و نتیجه جمع را در آزمایشگاه مجازی تجربه کنید. مدار شبیه سازی شده را به کلاس ارائه نمایید.

۴-۲-۳ تفریق کننده ناقص H. S

(Half Subtractor): عمل تفریق دودویی را در فصل

اول فرا گرفتیم و همان طور که اشاره شده برای اجرای عمل تفریق از متمم کامل اعداد استفاده می‌کنیم.

مدار تفریق کننده ناقص یا نیم تفریق کننده شامل

دو ورودی (x, y) و دو خروجی (B, D) است. خروجی D

حاصل تفریق و خروجی B رقم قرضی (Borrow) می‌باشد.

در حین تفریق (x y)، اگر $x \geq y$ باشد یعنی (۰-۰)،

۱-۰، ۱-۱ عمل تفریق انجام می‌شود و به رقم قرضی

نیاز نداریم. ولی اگر $x < y$ باشد یعنی (۰-۱) بایستی

یک واحد از مرتبه بالاتر قرض بگیریم. یک واحد قرض

گرفته شده از مرتبه بالاتر، ۲ واحد به بیت مورد نظر

اضافه می‌کند (در سیستم دودویی). این عمل را عیناً در

سیستم ده دهی شاهد هستیم که منجر به اضافه شدن

۱۰ واحد به رقم قرض گیرنده می‌شود. پس حاصل تفریق

۱۰-۱=۹ برابر با یک می‌شود. جدول ۴-۵ جدول

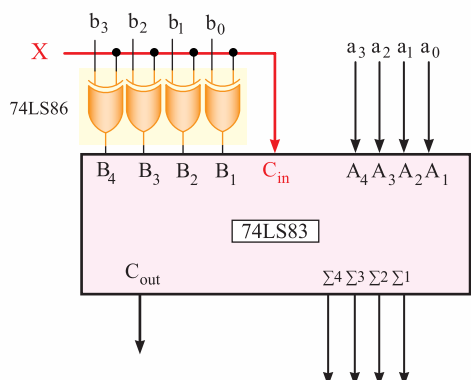
صحت تفریق کننده ناقص را نشان می‌دهد.

جدول ۴-۵ جدول صحت نیم تفریق کننده

ورودی‌ها	خروجی‌ها	می‌دانیم
X Y	B D	
0 0	0 0	0 - 0 = 0
0 1	1 1	0 - 1 = 1
1 0	0 1	1 - 0 = 1
1 1	0 0	1 - 1 = 0

همان‌طور که در شکل ۴-۲۰ مشاهده می‌کنید، چگونگی عمل تفریق دو عدد باینری چهار بیتی با استفاده از آی‌سی ۷۴۸۳ نشان داده شده است. ورودی C_{in} جمع‌کننده کامل به V_{CC} وصل شده است تا با افزودن «۱» به \bar{B} ، متمم ۲، B ایجاد شود.

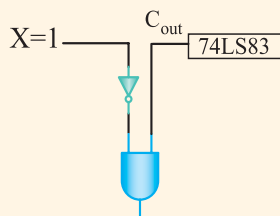
اگر مدار شکل ۴-۲۰ را به صورت شکل ۴-۲۱ تغییر دهیم، بسته به آن که خط کنترل X ، «۰» یا «۱» باشد، قادر به انجام عمل $(A+B)$ یا تفریق $(B-A)$ خواهد بود.



شکل ۴-۲۱- مدار جمع‌کننده / تفریق‌کننده چهار بیتی



نکته: ورودی C_{in} جمع‌کننده کامل به خط X وصل شده است وقتی $x=1$ است. (۱) واحد به \bar{B} اضافه می‌شود تا متمم (۲) عدد B به دست آید. برای حذف C_{out} در این مرحله از مدار شکل ۴-۲۲ استفاده می‌شود.



شکل ۴-۲۲- مدار مربوط به پایه C_{out} و خط X

مدار تفریق‌کننده کامل از دو مدار تفریق‌کننده ناقص و یک گیت OR مطابق شکل ۴-۱۹ تشکیل شده است. در عمل برای انجام دادن تفریق $(x-y)$ ، x را با مکمل دو y جمع می‌کنند و رقم نهایی را نادیده می‌گیرند. به مثال زیر توجه کنید.

مثال ۴-۴: تفریق مستقیم دو عدد باینری

$$\begin{array}{r} 11011 \\ - 01110 \\ \hline 01101 \end{array}$$

حاصل تفریق

اگر متمم دو عدد 01110 را به دست آوریم خواهیم داشت:

$$01110 \xrightarrow{\text{متمم ۲}} 10010$$

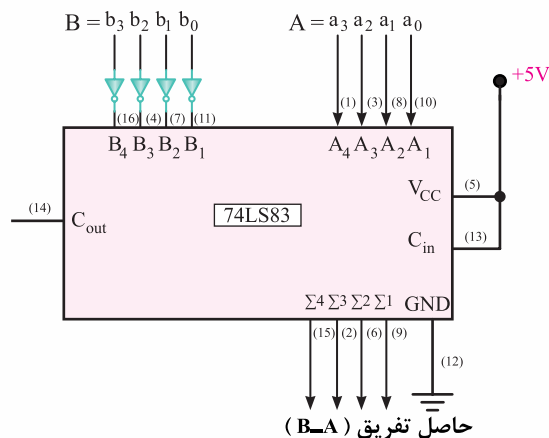
حال عدد 11011 را با متمم ۲ جمع می‌کنیم. رقم نقلی نهایی را حذف می‌کنیم. همان‌طور که مشاهده می‌شود حاصل تفریق در این روش مشابه حاصل تفریق در روش مستقیم است.

$$\begin{array}{r} 11011 \\ + 10010 \\ \hline \cancel{0}1101 \end{array}$$

حاصل تفریق

رقم نقلی نهایی

در شکل ۴-۲۰ مدار تفریق‌کننده را با استفاده از آی‌سی ۷۴۸۳ و به کمک متمم ۲ مشاهده می‌کنید.



شکل ۴-۲۰- انجام دادن تفریق به کمک متمم ۲



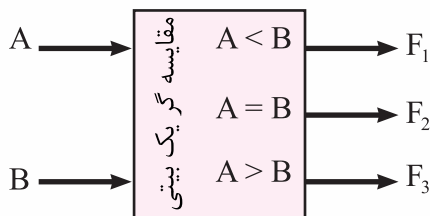
تمرین کلاسی ۴-۴: چه تفاوتی در خروجی گیت X-NOR و گیت تمرین کلاسی ۳ وجود دارد؟ بنویسید و جدول صحت آن را نیز رسم کنید.

در مقایسه بین دو بیت، ممکن است بزرگ‌تر، کوچک‌تر و مساوی بودن بیت‌ها مورد نظر باشد. جدول ۴-۷ جدول مقایسه بین دو بیت A و B را نشان می‌دهد.

جدول ۴-۷- جدول مقایسه دو بیت A و B

A	B	F_1 $A < B$	F_2 $A = B$	F_3 $A > B$
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

مطابق جدول ۴-۷ سه خروجی داریم. در شکل ۴-۲۳ بلوک دیاگرام مقایسه‌کننده یک بیتی را مشاهده می‌کنید:



شکل ۴-۲۳- بلوک دیاگرام مقایسه‌گر یک بیتی

با توجه به جدول صحت مقایسه‌گر یک‌بیتی، عبارت بولی هریک از خروجی‌های F_1 ، F_2 و F_3 را می‌نویسیم:



نکته: L به معنی Lower (کمتر)، E به معنی Equal (معاذل) و G به معنی Greater (بزرگ‌تر) است.



جهت هنجاریان علاقه‌مند: در آزمایشگاه با استفاده از یک عدد آی‌سی ۷۴۸۳ مدار شکل‌های ۴-۱۶ و ۴-۱۷ را ببینید و با تغییر ورودی‌ها به حالت‌های صفر و یک نتیجه عمل جمع یا تفریق دو عدد دل‌خواه را مشاهده کنید و جدول صحت مدار را رسم کنید.

مربیان عزیز با استفاده از نرم‌افزار مولتی سیم، مدار تفریق‌کننده کامل ۴ بیتی را ببینید و نتیجه تفریق را در آزمایشگاه مجازی و مدار شبیه‌سازی شده به کلاس ارائه نمایید.

۴-۲-۵- مقایسه‌کننده یک‌بیتی: درمقایسه دو بیت، اگر فقط قرار باشد که تساوی یا عدم تساوی نشان داده شود، ساده‌ترین مدار استفاده از گیت X-NOR است. در این مقایسه‌کننده اگر دو بیت مساوی باشند خروجی «۱» و در غیر این صورت خروجی «۰» می‌شود. جدول ۴-۶، جدول صحت گیت X-NOR را نشان می‌دهد.

جدول ۴-۶- جدول صحت گیت X-NOR

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1



تمرین کلاسی ۴-۳: از چه گیت دیگری می‌توان برای مقایسه مساوی بودن یا مساوی نبودن دو بیت استفاده کرد؟ جدول صحت آن را نیز رسم کنید.

- اطلاعات اول: به صورت کد شده ← ۰۰
- اطلاعات دوم: به صورت کد شده ← ۰۱
- اطلاعات سوم: به صورت کد شده ← ۱۰
- اطلاعات چهارم: به صورت کد شده ← ۱۱

چون هر یک از اطلاعات دارای کدی متفاوت با بقیه است، هیچ کدام از کدها به اشتباه به جای دیگری استفاده نمی‌شود. بنابراین برای رمز کردن کافی است که بتوانیم اطلاعات را به صورت رشته‌هایی متفاوت از صفرها و یک‌ها درآوریم.

اگر اطلاعاتی که به صورت کد درآورده می‌شوند، تنها اعداد باشند، کدهای به دست آمده را کدهای عددی و اگر اطلاعاتی که به صورت کد درآورده می‌شوند، حروف الفبا، ارقام یا علائم باشند، کدهای حرفی عددی نامیده می‌شوند. به طور کلی اگر یک برنامه نرم‌افزاری نصب شده در یک بانک را در نظر بگیریم، مشاهده می‌کنیم که جستجوی اطلاعات از دو طریق شماره حساب و نام فرد امکان‌پذیر است.

در نتیجه در این سامانه به هر دو نوع کد عددی و حرفی نیاز داریم. کدهای متنوعی برای نمایش اعداد تعریف شده است، که هر کدام اهداف خاصی را دنبال می‌کنند. ولی به طور کلی می‌توان انواع کد عددی را به صورت جدول ۴-۸ تقسیم‌بندی کرد.

جدول ۴-۸- انواع کد عددی

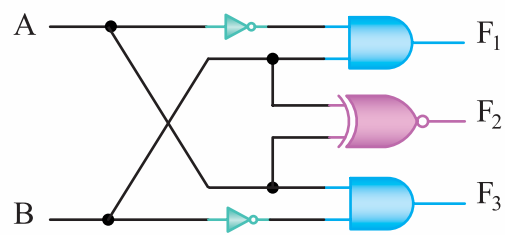
۱- کد وزن دار <div> { <div>الف وزن دار مثبت</div> <div>ب وزن دار منفی</div> </div>	
۲- کد بدون وزن <div> { <div>الف همینگ</div> <div>ب گری</div> </div>	پ BCD ۱- وزن ۴۲۲۱ ۲- وزن ۸۴۲۱
۳- کد باینری (دودویی)	
۴- کد اکتال (هشت تایی)	
۵- کد هگزا دسی مال (شانزده تایی)	

$$F_1 = \bar{A}B$$

$$F_2 = \bar{A}\bar{B} + AB$$

$$F_3 = A\bar{B}$$

با استفاده از توابع به دست آمده از خروجی‌ها می‌توان مدار شکل ۴-۲۴ را طراحی کرد.

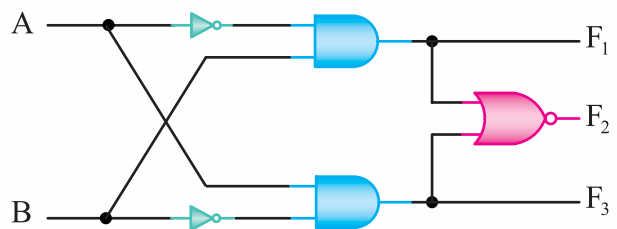


شکل ۴-۲۴- مدار مقایسه گر یک بیتی



تمرین کلاسی ۴-۵: مدار ۴-۲۵ را تحلیل کنید

و جدول صحت هر یک از خروجی‌های F_1 ، F_2 و F_3 را رسم کنید. نتیجه را با مدار شکل ۴-۲۴ مقایسه کنید.



شکل ۴-۲۵- مدار تمرین کلاسی

۴-۳- انواع کدها

از آن جا که رمز کردن اطلاعات برای ایجاد ارتباط با رایانه صورت می‌گیرد و از طرفی رایانه تنها صفرها و یک‌ها را می‌شناسد، برای کد کردن اطلاعات کافی است آنها را به صورت رشته‌ای از صفرها و یک‌ها درآوریم. که مثلاً برای نشان دادن کد مربوط به چهار نوع اطلاعات دوبیتی متفاوت، به صورت زیر عمل می‌کنیم:

توجه داشته باشید که در این روش نمایش اعداد، باید هر رقم دهدهی را با چهار بیت باینری نمایش دهیم. در جدول ۹-۴ تفاوت نمایش ارقام دهدهی از صفر تا ۹ به صورت باینری و BCD نمایش داده شده است.

جدول ۹-۴ نمایش ارقام دهدهی ۰ تا ۹ به صورت باینری

BCD و

عدد دهدهی	عدد باینری	عدد BCD
0	0	0000
1	1	0001
2	10	0010
3	11	0011
4	100	0100
5	101	0101
6	110	0110
7	111	0111
8	1000	1000
9	1001	1001



تمرین کلاسی ۶-۴: اعداد ۷۱۵ و ۴۶۰ در سیستم دهدهی را به صورت باینری و کد BCD به دست آورید.

با توجه به تبدیل سیستم‌های باینری، اکتال و هگزادسی‌مال به یکدیگر، می‌توان انواع کدها را به کد BCD تبدیل کرد.

۴-۴-۴ مبدل BCD به 7.S (نمایشگر هفت قطعه‌ای)

نمایشگر هفت قسمتی 7.S (Seven segment display) برای نمایش هر یک از ارقام ۰ تا ۹ به کار می‌رود و با توجه به آن چه که توضیح داده می‌شود، معمولاً رقم دهدهی را می‌توان به راحتی به کد BCD تبدیل کرد.

در فصل اول به تفصیل در مورد کدهایی که استفاده بیشتری دارند توضیح داده شده است و معرفی این کدها جهت آشنایی است.

۱-۳-۴- مقایسه انواع کد با کد BCD: بعضی از ماشین‌های محاسبه‌گر الکترونیکی عملیات ریاضی را در کد BCD انجام می‌دهند. در کد BCD هر رقم دهدهی با چهار بیت باینری معادل آن نشان داده می‌شود.

مثال ۵-۴: اعداد ۳ و ۹ و ۲۳۵ در سیستم دهدهی را به صورت کد به دست آورید.

حل: برای تبدیل اعداد باینری به کد BCD در صورتی که تعداد ارقام آنها کمتر از ۴ رقم باشد. کد باینری را می‌نویسیم و سپس به تعدادی که بیت کم دارد در سمت چپ کد باینری صفر اضافه می‌کنیم. مثلاً برای عدد $(3)_{10}$ داریم:

$$(3)_{10} = (11)_2 = 0011$$

معادل باینری ۳

تعداد صفرهای اضافه شده
کد BCD

در صورتی که تعداد ارقام باینری در جواب عدد مورد نظر برابر با ۴ رقم باشد همان را می‌نویسیم. مثلاً برای عدد $(9)_{10}$ داریم:

$$(9)_{10} = (1001)_2 = (1001)_{BCD}$$

توجه شود اگر عدد دهدهی چند رقمی باشد، برای هر رقم دهدهی، کد باینری را به دست می‌آوریم، سپس معادل آن را به صورت BCD می‌نویسیم. مثلاً برای عدد $(235)_{10}$ داریم:

$$(235)_{10} = (11101011)_2$$

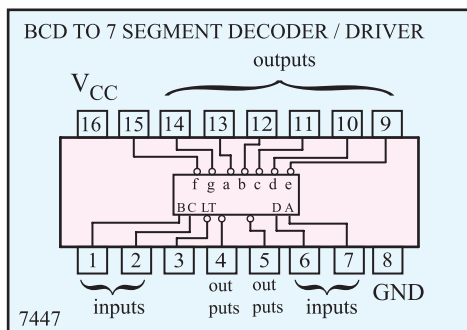
معادل هر رقم دهدهی در سیستم باینری $(101)_2$ $(11)_2$ $(10)_2$

معادل هر رقم باینری در سیستم BCD (0101) (0011) (0010)

کد BCD

$$(235)_{10} = (11101011)_2 = (001000110101)_{BCD}$$

آی سی پایه های a, b, c, d, e, f و g هستند که به ترتیب به پایه های ۱۳، ۱۲، ۱۱، ۱۰، ۹، ۱۵ و ۴ وصل می شوند. پایه های ۳، ۴ و ۵ آی سی را به V_{CC} وصل می کنند.



شکل ۲۷-۴ نقشه داخلی آی سی ۷۴۴۷



جهت هنرجویان علاقه مند: مدار BCD به 7.S را برای آی سی ۷۴۴۸ (کاتد مشترک) توسط نرم افزار مولتی سیم ببندید و نتیجه فعالیت آزمایشگاهی را به کلاس ارائه کنید.

۴-۴-۱- مبدل هگزادسی مال به 7.S: در سیستم هگزادسی مال (شانزده تایی) ارقام از صفر شروع می شوند و تا ۱۵ خاتمه می یابند. همان طور که می دانید از رقم ۱۰ تا ۱۵ را در این سیستم به صورت حروف A تا F نمایش می دهند.

در مبدل هگزادسی مال به 7.S برای نمایش ارقام می توان از جدول ۴-۱۰ استفاده کرد. البته اگر بخواهیم ارقام سیستم هگزادسی مال را به صورت BCD در 7.S نمایش دهیم باید از دو عدد 7.S استفاده کنیم، (آیا می دانید چرا؟)

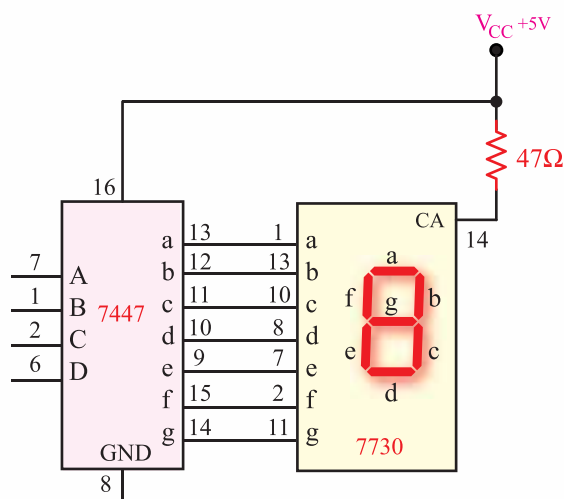
چون هدف در این قسمت مبدل هگزادسی مال به 7.S است فقط از یک آی سی ۷۴۴۷ و یک 7.S استفاده می کنیم و مداری مشابه مدار ۴-۲۶ را می بندیم. ورودی ها مانند مدار ۴-۲۶ در پایه های A, B, C و D آی سی ۷۴۴۷ قرار دارند. جدول ۴-۱۰ اعداد صفر تا ۱۵ در سیستم ده دهی، تبدیل

برای نمایش کد BCD در 7.S باید از آی سی ۷۴۴۷ (برای 7.S آند مشترک) و آی سی ۷۴۴۸ (برای 7.S کاتد مشترک) استفاده کنیم.

شکل ۲۶-۴ مدار مربوط به این مبدل را نشان می دهد. رقم BCD را به ورودی های آی سی ۷۴۴۷ می دهیم. پایه های A, B, C و D ورودی های آی سی هستند و پایه های a, b, c, d, e, f و g خروجی های آی سی که به 7.S اتصال می یابند.

ورودی D با ارزش ترین رقم BCD و ورودی A کم ارزش ترین رقم BCD را دریافت می کند.

آی سی ۷۷۳۰ نمایشگر هفت قسمتی آند مشترک است. پایه ۱۴ آند مشترک (CA) و ورودی مشترک برای تمام LED ها است.

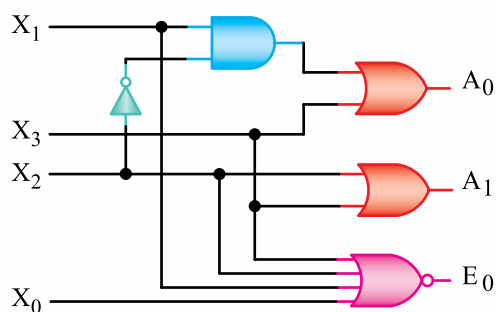


شکل ۲۶-۴ مبدل BCD به 7.S

برای محدود کردن جریان و جلوگیری از آسیب رسیدن به نمایشگر هفت قطعه ای 7.S یک مقاومت 47Ω را به صورت سری با V_{CC} و در مسیر پایه مشترک ۱۴ قرار می دهیم.

در شکل ۲۷-۴ نقشه داخلی و پایه های آی سی ۷۴۴۷ را ملاحظه می کنید.

V_{CC} به پایه ۱۶ و زمین به پایه ۸ آی سی وصل می شود. پایه های ورودی A, B, C و D به ترتیب به پایه های ۷، ۶، ۵ و ۴ آی سی متصل می شود. خروجی های



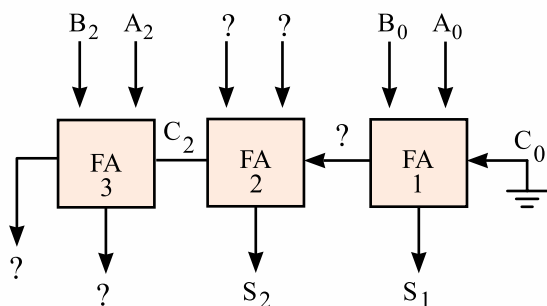
شکل ۲۸-۴ مدار سؤال ۲

جدول صحت سؤال ۲ الگوی پرسش

X_3	X_2	X_1	X_0	A_1	A_0	E_0
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

۳- یک مقایسه کننده طراحی کنید که در آن دو عدد دو بیتی A و B را با هم مقایسه کند و وضعیت $A=B$ را در خروجی نشان دهد.

۴- بلوک دیاگرام شکل ۲۹-۴ را که مربوط به یک جمع کننده ۳ بیتی با استفاده از تمام جمع کننده است را کامل کنید.



شکل ۲۹-۴ مدار سؤال ۴

آن به هگزادسی مال و نمایش آن در 7.S را نشان می دهد.

جدول ۱۰-۴- مبدل هگزا دسی مال به 7.S و نمایش ارقام

نمایش عدد	پایه های 7-Seg a b c d e f g	کد هگزا دسی مال	کد باینری	سطر
0	1 1 1 1 1 1 0	0	0 0 0 0	0
1	0 1 1 0 0 0 0	1	0 0 0 1	1
2	1 1 0 1 1 0 1	2	0 0 1 0	2
3	1 1 1 1 0 0 1	3	0 0 1 1	3
4	0 1 1 0 0 1 1	4	0 1 0 0	4
5	1 0 1 1 0 1 1	5	1 0 1 0	5
6	1 0 1 1 1 1 1	6	0 1 1 0	6
7	1 1 1 0 0 0 0	7	0 1 1 1	7
8	1 1 1 1 1 1 1	8	1 0 0 0	8
9	1 1 1 1 0 1 1	9	1 0 0 1	9
A	1 1 1 1 1 0 1	A	1 0 1 0	10
B	0 0 1 1 1 1 1	B	1 0 1 1	11
C	1 0 0 1 1 1 0	C	1 1 0 0	12
D	0 1 1 1 1 0 1	D	1 1 0 1	13
E	1 1 0 1 1 1 1	E	1 1 1 0	14
F	1 0 0 0 1 1 1	F	1 1 1 1	15

۵-۴- الگوی پرسش

۱- نشان دهید چگونه می توان سه تابع:

$$F_1 = A \oplus B \oplus C$$

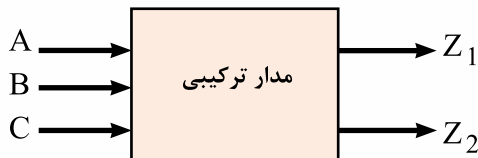
$$F_2 = \bar{A}BC + A\bar{B}C$$

$$F_3 = ABC$$

را به کمک سه مدار نیم جمع کننده اجرا کرد .

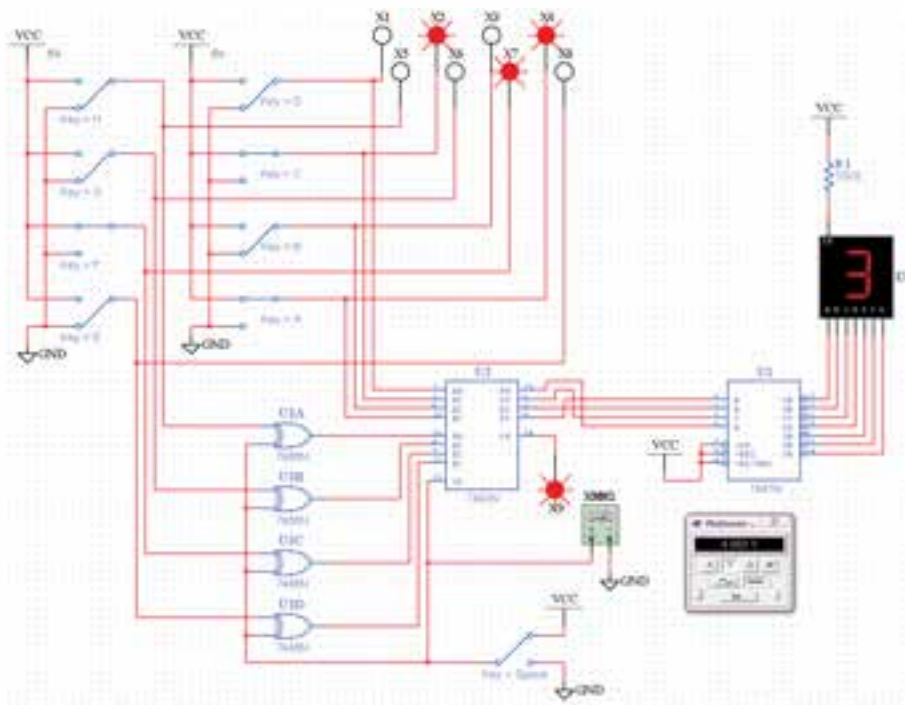
۲- جدول صحت مدار شکل ۲۸-۴ را به دست آورید.

۵- مدار ترکیبی را رسم کنید و جدول صحت آن را به دست آورید.



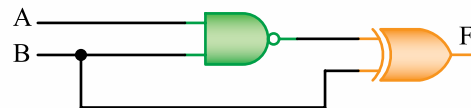
شکل ۳۱-۴ مدار سؤال ۷

۸- مدار شکل ۳۲-۴ مدار جمع کننده و تفریق کننده با استفاده از آی سی ۷۴۸۳ است. این مدار را با استفاده از نرم افزار ببندید و به کلاس ارائه نمائید. کلید $\text{key} = \text{Space}$ را اگر به زمین وصل کنید مدار به عنوان جمع کننده ۴ بیتی عمل می کند و می توانید با تغییر کلیدهای A تا H اعداد مختلف ۴ بیتی (باینری) را به مدار اعمال کنید. و نتیجه جمع را در 7.S مشاهده کنید. چنانچه $\text{key} = \text{Space}$ را به V_{CC} وصل کنید مدار به عنوان تفریق کننده کامل ۴ بیتی عمل می کند. با تغییر کلیدهای ورودی، نتیجه را در 7.S مشاهده کنید.



شکل ۳۲-۴ مدار جمع کننده و تفریق کننده

۵- در شکل ۳۰-۴ تابع خروجی را بنویسید و جدول صحت آن را به دست آورید.



شکل ۳۰-۴ مدار سؤال ۵

۶- دو عدد $A = 10101110$ و $B = 10011111$ را با هم جمع کنید و حاصل جمع را در یک مدار تمام جمع کننده با استفاده از آی سی ۷۴ LS ۸۳ نشان دهید. آیا پایه ۱۴ آی سی دوم خروجی یک دارد؟
۷- در مدار شکل ۳۱-۴ شرایط زیر برقرار است:
(الف) موقعی یک است که $C=0$, $A=B=1$ یا $B=0$ و $A=C=1$ باشد.
(ب) موقعی یک است که $C=1$, $B=1$ و $A=0$ یا $C=1$ و $B=0$ و $A=1$ باشد.
(ج) در بقیه حالات $Z_1=Z_2=0$ است. مدار منطقی این



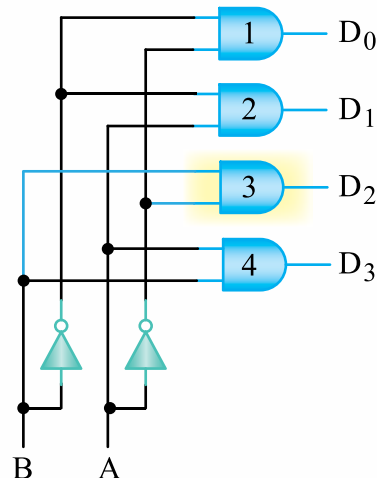
جهت هنرجویان علاقه مند:

۹- جدول صحت تفریق کننده کامل را رسم کنید.

۴-۶- مدارهای رمزگشا (Decoder):

برای دریافت اطلاعات از دستگاه‌های محاسباتی دیجیتال مداری مورد نیاز است که اطلاعات را از حالت دودویی به اعشاری تبدیل کند. خروجی این مدارها معمولاً به نمایشگرها متصل می‌شود. این تبدیل‌کننده‌ها را رمزگشا و عملی را که انجام می‌دهند رمزگشایی می‌نامیم. هر رمزگشا با n ورودی دارای حداکثر 2^n خروجی است و در هر لحظه تنها یکی از 2^n خروجی فعال است. به عبارت دیگر، هر یک از خروجی‌های آن متناظر با یک ترکیب خاص ورودی (یک جمله حاصل ضرب نرمال یا مین‌ترم) است.

در شکل ۴-۳۳ یک رمزگشای $2 \rightarrow 4$ (بخوانید ۲ به ۴) و در جدول ۴-۱۱ جدول صحت آن نشان داده شده است.



شکل ۴-۳۳- مدار رمزگشای $2 \rightarrow 4$

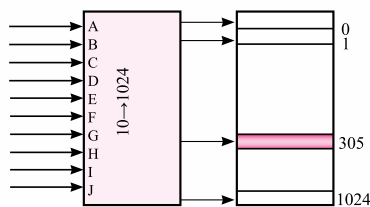
جدول ۴-۱۱- جدول صحت مدار رمزگشا

B	A	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

ورودی‌های A و B را ورودی‌های آدرس و خروجی‌های D_0 , D_1 , D_2 و D_3 را خروجی‌های داده می‌نامیم. همان‌طور که در شکل دیده می‌شود، خروجی دروازه

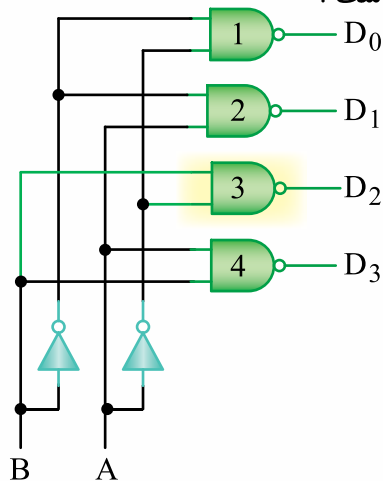
شماره ۱ فقط به ازای ترکیب ورودی $BA = 00$ فعال می‌شود («۱» می‌شود)؛ یعنی: $D_0 = \bar{A}\bar{B}$. به همین ترتیب، می‌توانیم بنویسیم: $D_1 = \bar{A}B$ و $D_3 = AB$.

(توجه کنید که در این جا A متغیر کم‌ارزش‌تر و B متغیر با ارزش‌تر از آن است، یعنی: $A=2^1=2$, $B=2^0=1$). از رمزگشاها برای آدرس‌دهی اجزای مختلف گیرنده یا فرستنده اطلاعات یک سیستم نیز استفاده می‌شود؛ مثلاً اگر بخواهیم کلمه‌ای را در یک سطر معین حافظه بنویسیم یا آن را از سطر معینی از حافظه بخوانیم، نخست باید محل سطر مورد نظر را مشخص کنیم. در شکل ۴-۳۴ سطر ۳۰۵ از یک حافظه با ظرفیت ۱۰۲۴ کلمه آدرس داده شده است (در بخش مدارهای ترتیبی با ساختمان حافظه‌های نیمه‌هادی آشنا خواهید شد).



شکل ۴-۳۴- آدرس دهی مکان یک کلمه معین از حافظه

ممکن است رمزگشا با دروازه‌های NAND ساخته شده باشد. در این صورت، حالت فعال خروجی‌ها «۰» خواهد بود. در شکل ۴-۳۵ یک رمزگشای $2 \rightarrow 4$ نشان داده شده است.

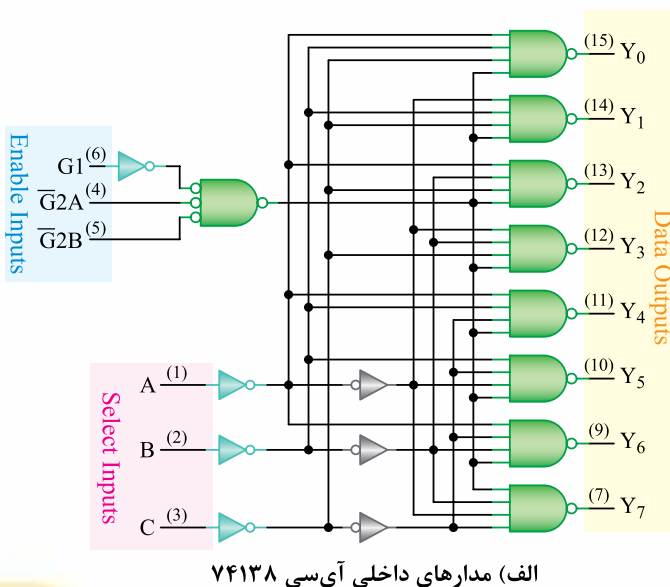


شکل ۴-۳۵- رمزگشای $2 \rightarrow 4$



نکته: در آی سی ها از کلمات Enable و Disable استفاده می کنند. کلمه Enable به معنی فعال کننده و Disable به معنی غیر فعال کننده به کار می رود.

با توجه به جدول صحت ۴-۱۳ هرگاه یک ورودی را با X نشان دهند به معنای این است که اگر ارزش منطقی صفر یا یک باشد برای خروجی مدار بی تفاوت است. و در سطر اول جدول صحت چون ورودی E در صفر منطقی قرار گرفته است عمل رمز گشایی انجام نمی شود. در سطرهای دوم تا پنجم چون ورودی E برابر با «یک» است عمل رمز گشایی انجام خواهد شد. در شکل ۴-۳۷ مدار یک رمز گشای ۸ → ۳ به همراه جدول صحت آن، جدول ۴-۱۴ نشان داده شده است. همان طور که ملاحظه می کنید، رمز گشا دارای دو خط فعال کننده G_1 و G_2 است که حالت فعال آنها برای G_1 برابر یک و برای G_2 برابر صفر است. در کتاب های راهنمای تراشه های تجارتي؛ وضعیت ورودی و خروجی ها را به جای صفر و یک با حروف (L مخفف Low) و (H مخفف High) مشخص می کنند.



جدول ۴-۱۲- جدول صحت رمز گشای ۴ → ۲

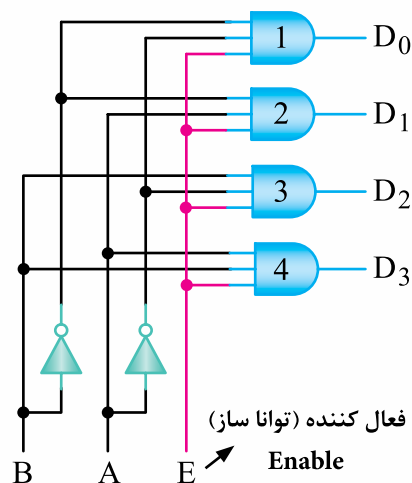
B	A	D ₀	D ₁	D ₂	D ₃
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

جدول ۴-۱۲- جدول صحت دکودر ۴ → ۲ با حالت فعال Low را نشان می دهد.

همان طور که در شکل نشان داده شده، دروازه شماره ۳ در حالت فعال است، ورودی های این دروازه از \bar{A} و B گرفته شده است.

در بعضی از رمز گشاها علاوه بر ورودی های آدرس، یک ورودی فعال کننده (Enable) (تواناساز) نیز پیش بینی شده است. اگر این ورودی در حالت غیر فعال نگه داشته شود، رمز گشایی انجام نخواهد شد.

در شکل ۴-۳۶ یک رمز گشا ۴ → ۲ با خط تواناساز به همراه جدول صحت آن را در جدول ۴-۱۳ مشاهده می کنید.



شکل ۴-۳۶- رمز گشای ۴ → ۲ با خط تواناساز

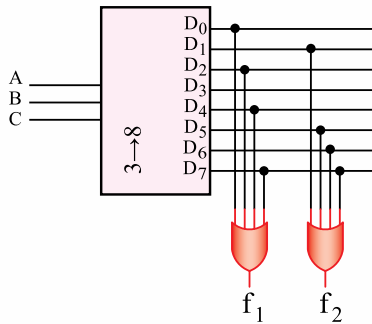
جدول ۴-۱۳- جدول صحت دکودر ۴ → ۲ با خط تواناساز

E	B	A	D ₀	D ₁	D ₂	D ₃
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

جدول ۴-۱۵- مربوط به مثال ۴-۶

	C	B	A	f_1	f_2
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

حل: چون $F_1 = \sum m(0, 2, 4, 7)$ و $F_2 = \sum m(1, 5, 6, 7)$ است، برای اجرای هریک از توابع F_1 و F_2 از یک دروازه OR چهار ورودی مطابق شکل ۴-۳۸ استفاده می‌کنیم.

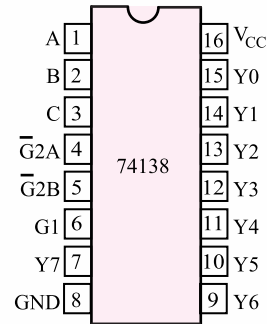


شکل ۴-۳۸- رمز گشای ۸ → ۳

اگر رمز گشا با دروازه‌های NAND ساخته شده باشد، در اجرای توابع منطقی به جای دروازه‌های OR باید از دروازه‌های NAND استفاده کنیم (به یاد آورید که ترکیب AND-OR معادل ترکیب NAND NAND است).

مثال ۴-۷: یک مدار تمام‌جمع‌کننده یک بیتی با استفاده از رمز گشا طراحی کنید.

حل: ابتدا جدول صحت S و C مربوط به جمع‌کننده کامل را به دست می‌آوریم حال با توجه به جدول صحت ۴-۱۶ داریم:



(ب) مشخصات پایه‌های آی‌سی ۷۴۱۳۸
شکل ۴-۳۷- مدارهای داخلی و مشخصات آی‌سی ۷۴۱۳۸



نکته: هدف از نمایش مدارهای داخلی آی‌سی فقط شناسایی پایه‌های Enable آی‌سی است و در آزمون‌ها از مدار داخلی آی‌سی نباید سؤال داده شود.

جدول ۴-۱۴- جدول صحت رمز گشای ۸ → ۳ با دو خط تواناساز

INPUTS												
ENABLE	SELECT	OUTPUTS										
G1	$\overline{G2}$	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

برای اجرای توابع منطقی نیز می‌توانیم از رمز گشا استفاده کنیم. بدین منظور باید هریک از متغیرها را به ورودی آدرس متناظر با ارزش آن و همه خروجی‌های رمز گشا را که متناظر با حالت‌های «۱» تابع است، به ورودی‌های یک دروازه OR وصل کنیم.

مثال ۴-۶: با توجه به جدول ۴-۱۵ توابع منطقی F_1 و F_2 را به کمک یک رمز گشای ۸ → ۳ اجرا کنید.

جدول ۱۶-۴: جدول صحت جمع کننده کامل

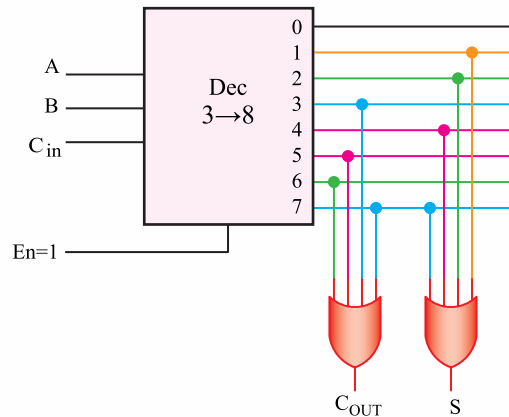
	A	B	C _{in}	C _{OUT}	S
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1

مطابق جدول ۱۷-۴ خروجی‌های S و C_{out} در سطرهای زیر جواب یک دارند.

$$S = \sum m(1, 2, 4, 7) \quad S = \sum m(1, 2, 4, 7)$$

$$C_{out} = \sum m(3, 5, 6, 7)$$

در تمام جمع کننده ۳ ورودی و ۲ خروجی داریم. پس رمزگشایی را انتخاب می‌کنیم که ۳ ورودی داشته باشد و ۸=۲^۳ خروجی و خروجی‌ها را بر اساس مین ترم‌ها به هم وصل می‌کنیم.



شکل ۳۹-۴: مدار جمع کننده با استفاده از رمزگشا

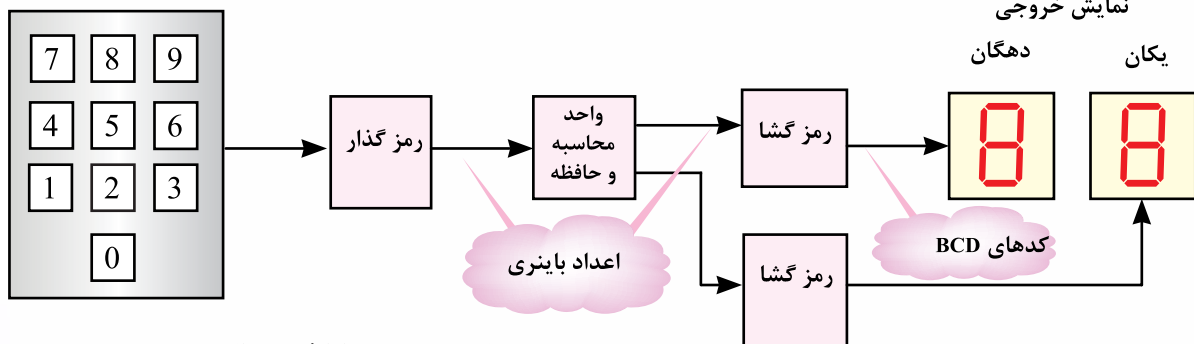
تمرین کلاسی ۷-۴: تابع منطقی رأی اکثریت یک کمیته سه نفره را با یک رمز گشای ۸→۳ اجرا کنید.

تمرین کلاسی ۸-۴: تابع $F = \sum m(1, 2, 4, 7)$ را با یک رمز گشای ۸→۳ که با گیت‌های NAND ساخته شده است را اجرا کنید.

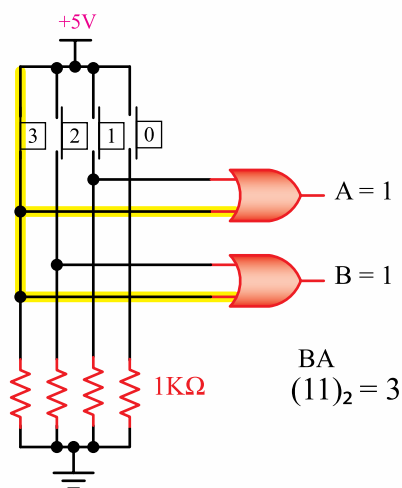
تمرین کلاسی ۹-۴: یک رمز گشای ۴→۲ با دروازه‌های AND را با خط تواناساز صفر فعال رسم کنید.

۷-۴- مدارهای رمز گذار (Encoder)

اعدادی که به کامپیوتر یا سامانه دیجیتالی داده می‌شود، در سیستم ددهی هستند. چون کامپیوتر با اعداد باینری کار می‌کند، اعداد ددهی باید به اعداد باینری تبدیل شوند. مداری که اطلاعات را از حالت ددهی به باینری تبدیل می‌کند، رمز گذار نام دارد. شکل ۴۰-۴ یک سامانه دیجیتالی را نشان می‌دهد.



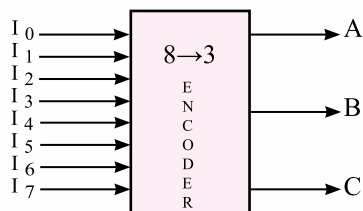
شکل ۴۰-۴: یک سامانه دیجیتالی



شکل ۴-۴۲ مدار رمزگذار ۲ → ۴

تمرین کلاسی ۴-۱۰: در مدار شکل ۴-۴۲ اعداد باینری صفر تا دو چگونه ظاهر می‌شوند؟ توضیح دهید.

در شکل ۴-۴۳ دیاگرام یک رمزگذار ۳ → ۸ (بخوانید ۸ به ۳) و در جدول ۴-۱۸ جدول صحت آن نشان داده شده است. همان‌طور که مشاهده می‌کنید، در هر لحظه باید فقط یکی از ورودی‌های رمزگذار در حالت فعال باشد.

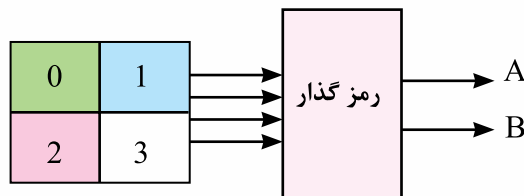


شکل ۴-۴۳ بلوک دیاگرام رمزگذار ۳ → ۸

جدول ۴-۱۸ جدول صحت یک رمزگذار ۳ → ۸

ورودی‌ها								خروجی‌ها		
I₇	I₆	I₅	I₄	I₃	I₂	I₁	I₀	C	B	A
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

یک رمزگذار با m ورودی ده‌دهی باید دارای n خروجی باشد؛ به‌طوری‌که همواره نامساوی $2^n \geq m$ برقرار باشد. برای مثال یک صفحه‌کلید با شماره‌های صفر تا سه را می‌توان به دو خط باینری تبدیل کرد. در شکل ۴-۴۱ بلوک دیاگرام یک رمزگذار ۲ → ۴ به همراه جدول صحت در جدول ۴-۱۷ نشان داده شده است.



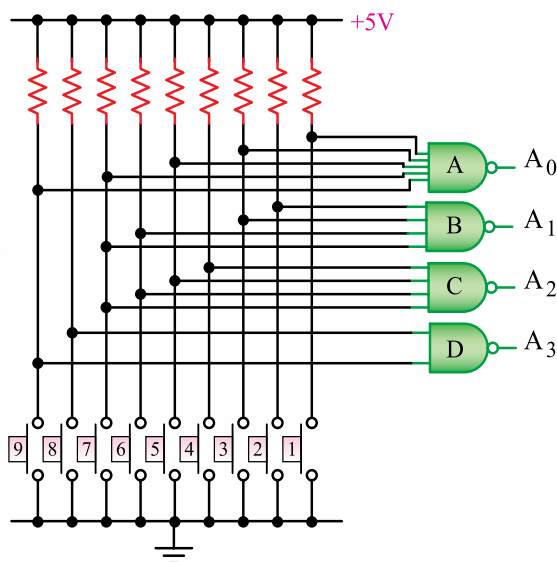
شکل ۴-۴۱ بلوک دیاگرام رمزگذار ۲ → ۴

جدول ۴-۱۷ جدول صحت رمزگذار ۲ → ۴

I₃	I₂	I₁	I₀	B	A
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

همان‌طور که مشاهده می‌کنید، در هر لحظه باید فقط یکی از ورودی‌های رمزگذار در حالت فعال قرار گیرد تا درست عمل کند. ورودی I_0, I_1, \dots معرّف کلیدهای ورودی است و ۱ در جدول نشان‌دهنده فعال بودن آن کلید است.

مدار رمزگذار ۲ → ۴ را می‌توان مطابق شکل ۴-۴۲ طراحی کرد. چنان‌چه کلید ۳ فشرده شود یکی از خطوط دو گیت «OR» منطقی می‌شود. که سبب ایجاد عدد باینری سه «۱۱» در خروجی رمزگذار می‌شود.



شکل ۴-۴۵- رمزگذار ده‌دهی به باینری

جدول ۴-۱۹- جدول صحت رمزگذار ده‌دهی به باینری

INPUTS									OUTPUTS			
1	2	3	4	5	6	7	8	9	A ₃	A ₂	A ₁	A ₀
0	1	1	1	1	1	1	1	1	0	0	0	1
1	0	1	1	1	1	1	1	1	0	0	1	0
1	1	0	1	1	1	1	1	1	0	0	1	1
1	1	1	0	1	1	1	1	1	0	1	0	0
1	1	1	1	0	1	1	1	1	0	1	0	1
1	1	1	1	1	0	1	1	1	0	1	1	0
1	1	1	1	1	1	0	1	1	0	1	1	1
1	1	1	1	1	1	1	0	1	1	0	0	0
1	1	1	1	1	1	1	1	0	1	0	0	1
1	1	1	1	1	1	1	1	1	0	0	0	0

همان‌گونه که متذکر شدیم، در یک لحظه معین فقط یکی از ورودی‌های رمزگذار می‌تواند فعال شود؛ مثلاً در شکل ۴-۴۵ اگر کلید ۹ فشرده شود، در خروجی مدار ترکیب باینری هم‌ارز آن یعنی ۱۰۰۱ ظاهر می‌شود.

حال اگر به طور اتفاقی کلیدهای ۶ و ۹ هم‌زمان

فشرده شوند، در خروجی مدار ترکیب باینری ۱۱۱۱ ظاهر می‌شود که معادل هیچ کدام از ارقام ۶ یا ۹ نیست. برای پیش‌گیری از بروز این‌گونه خطاها، رمزگذارهای صنعتی

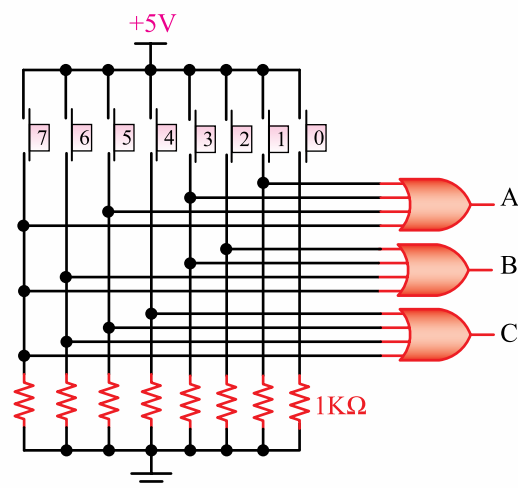
مطابق جدول فوق، معادلات بول هر یک از خروجی‌های A، B و C را می‌نویسیم:

$$A = I_1 + I_3 + I_5 + I_7$$

$$B = I_1 + I_3 + I_6 + I_7$$

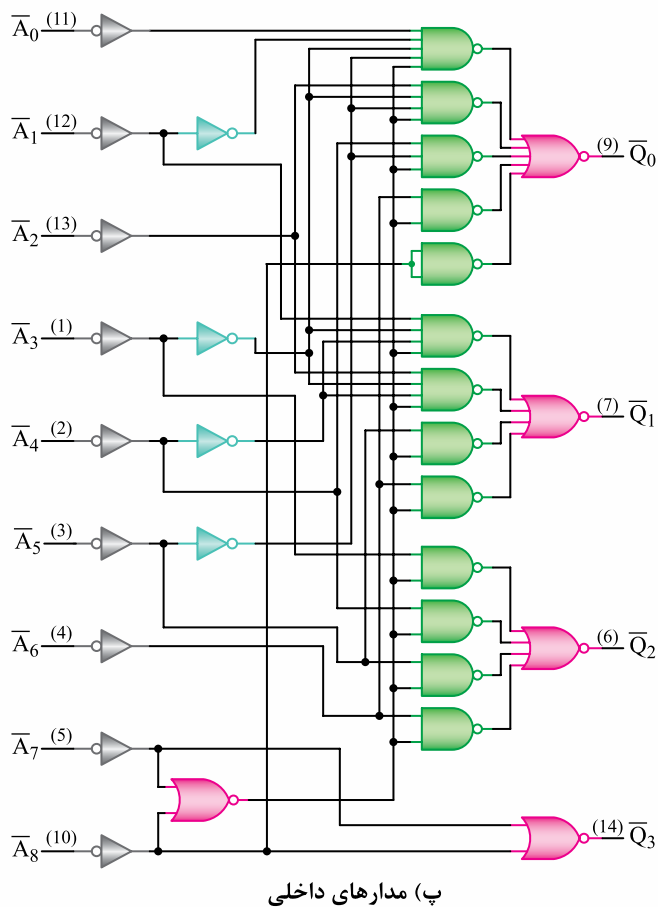
$$C = I_1 + I_5 + I_6 + I_7$$

مدار این رمزگذار در شکل ۴-۴۴ رسم شده است.



شکل ۴-۴۴- مدار رمزگذار ۳ → ۸

مدارهای رمزگذار را می‌توان با دروازه‌های منطقی NAND نیز طراحی کرد. در این صورت، حالت فعال ورودی‌ها «۰» خواهد بود. در شکل ۴-۴۵ مدار یک رمزگذار ده‌دهی به باینری که با دروازه‌های NAND اجرا شده است را مشاهده می‌کنید. جدول صحت این مدار را در جدول ۴-۱۹ آورده‌ایم. توجه کنید که در این مدار برای رقم صفر، ورودی خاصی در نظر گرفته نشده است؛ زیرا زمانی که همه کلیدهای ۱ تا ۹ قطع باشند، خروجی‌های مدار در حالت «۰۰۰۰» منطقی قرار می‌گیرند که معادل صفر اعشاری است.



شکل ۴-۴۶ رمزگذار تقدمی SNV4147

جدول ۴-۲۰ جدول صحت آی سی SNV4147 را نشان می دهد.

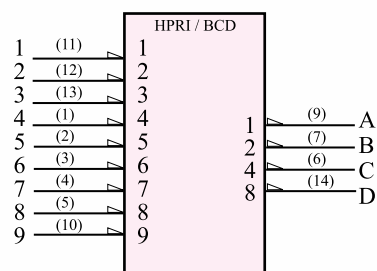
جدول ۴-۲۰- جدول صحت رمزگذار تقدمی SNV4147

INPUTS									OUTPUTS			
1	2	3	4	5	6	7	8	9	A ₃	A ₂	A ₁	A ₀
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

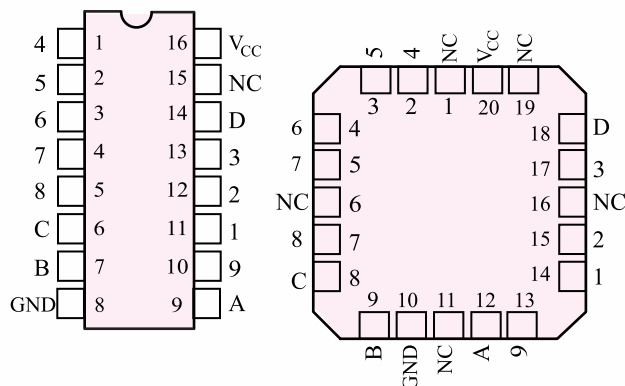
H = High logic level , L = Low logic level , X = irrelevant

را طوری طراحی می کنند که در صورت فشرده شدن همزمان چند کلید، فقط ترکیب باینری هم ارز بزرگترین رقم اعشاری را تولید کنند. این رمزگذارها را رمزگذارهای با رعایت تقدم (Priority Encoder) می نامند. مثلاً اگر دو کلید ۶ و ۹ همزمان فشرده شوند، رمزگذار صنعتی ارزش کلید ۹ را تولید می کند.

تراشه (آی سی) SNV4147 یک رمزگذار ده دهی به باینری با رعایت تقدم است. مشخصات کامل آن را در شکل ۴-۴۶ مشاهده می کنید. در شکل ۴-۴۶ الف علامت استاندارد آی سی، در شکل ۴-۴۶ ب نمای بالا و شماره پایه های آی سی و در شکل ۴-۴۶ پ مدارهای داخلی آن را ملاحظه کنید.



الف) علامت استاندارد IEEE/ANSI



ب) نمای بالا و شماره پایه ها

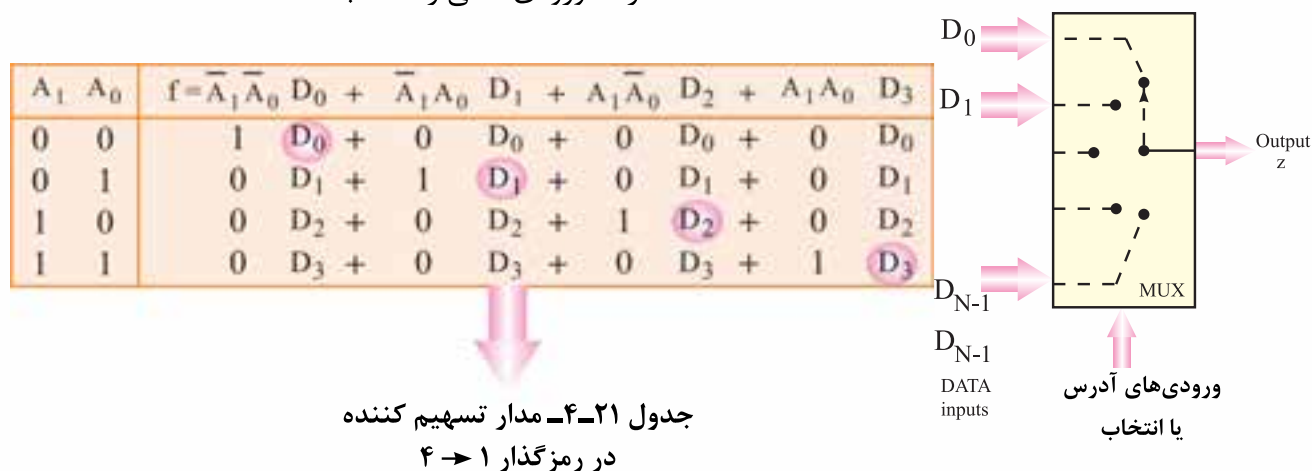
این مدار با توجه به آدرسی که برای آن انتخاب می‌شود، به یکی از ورودی‌ها اجازه عبور می‌دهد. به عبارت دیگر، عملکرد آن درست شبیه یک کلید چند حالتی است؛ با این تفاوت که حالت کلید به صورت دیجیتالی انتخاب می‌شود. دیاگرام عملیاتی یک مالتی پلکسر در شکل ۴-۴۷ نشان داده شده است. ورودی‌های انتخاب حالت را و ورودی‌های آدرس (Address Inputs) و ورودی‌های اصلی مالتی پلکسر ورودی‌های داده (Data inputs) و خروجی مالتی پلکسر را خروجی تابع یا به اختصار خروجی می‌نامند. یک مالتی پلکسر با n خط آدرس، می‌تواند یکی از 2^n ورودی اصلی را انتخاب کند.



تذکر مهم: هدف از نمایش مدارهای داخلی آی‌سی‌ها آشنایی با مدارهاست و به هیچ عنوان در آزمون‌ها نباید سؤالی در این رابطه طراحی شود.

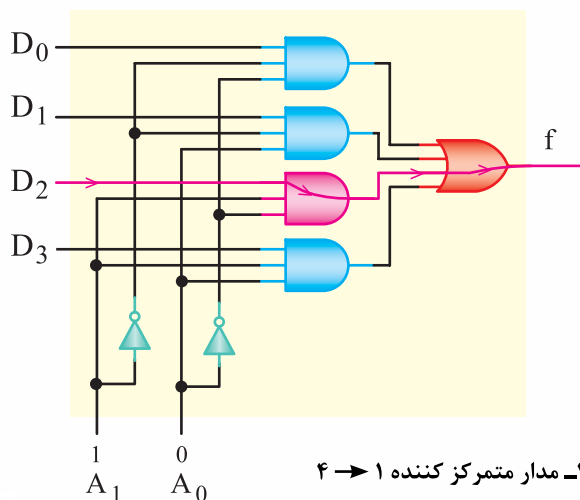
۴-۸- مدارهای متمرکز کننده یا تسهیم کننده (Multiplexer)

مالتی پلکسر یکی از پرکاربردترین مدارهای ترکیبی است که مانند یک انتخاب کننده (Selector) عمل می‌کند.



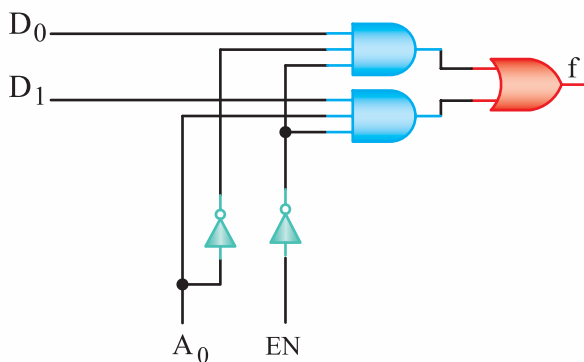
A_1	A_0	f
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

شکل ۴-۴۷- شمای عمومی یک متمرکز کننده



شکل ۴-۴۸- مدار متمرکز کننده ۱ → ۴

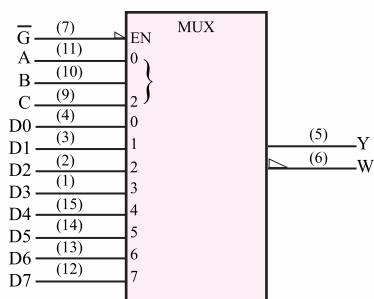
در شکل ۴-۴۸ مدار یک مالتی پلکسر ۱ → ۴ (بخوانید ۴ به ۱) نشان داده شده است. در این شکل، ورودی‌های اصلی مالتی پلکسر با D_0, D_1, D_2 و D_3 ورودی‌های آدرس آن با A_0 و A_1 و خروجی مدار با F مشخص شده است.



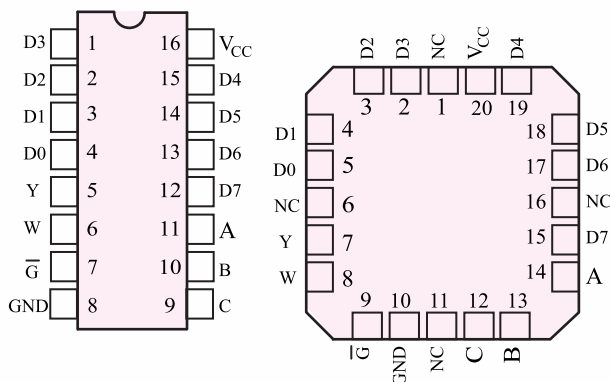
شکل ۴-۴۹- مدار یک مالتی پلکسر ۱ → ۲

در شکل ۴-۵۰ مدار یک مالتی پلکسر ۱ → ۸ را که با شماره تجارتي ۷۴۱۵۱ به بازار عرضه می شود مشاهده می کنید.

در شکل ۴-۵۰ الف نماد استاندارد آی سی، در شکل ۴-۵۰ ب نمای بالا و شماره پایه های آی سی و در شکل ۴-۵۰ پ مدارهای داخلی آی سی نشان داده شده است.



الف) نماد استاندارد IEEE/ANSI



ب) نمای بالا و شماره پایه

با توجه به شکل ۴-۴۸ معادله بولی تابع F را می نویسیم.

$$F = \bar{A}_1 \bar{A}_0 D_0 + \bar{A}_1 A_0 D_1 + A_1 \bar{A}_0 D_0 + A_1 A_0 D_1$$

تابع F به فرم مجموع حاصل ضربها است؛ یعنی، در هر حالت فقط یکی از جملات آن ممکن است «۱» باشد. مثلاً اگر حالت $A_1=0$ و $A_0=1$ را انتخاب کنیم، فقط جمله $A_1 \bar{A}_0$ «۱» است. لذا خواهیم داشت:

$$F = 0 \times D_0 + 0 \times D_1 + 1 \times D_0 + 0 \times D_1 = D_0$$

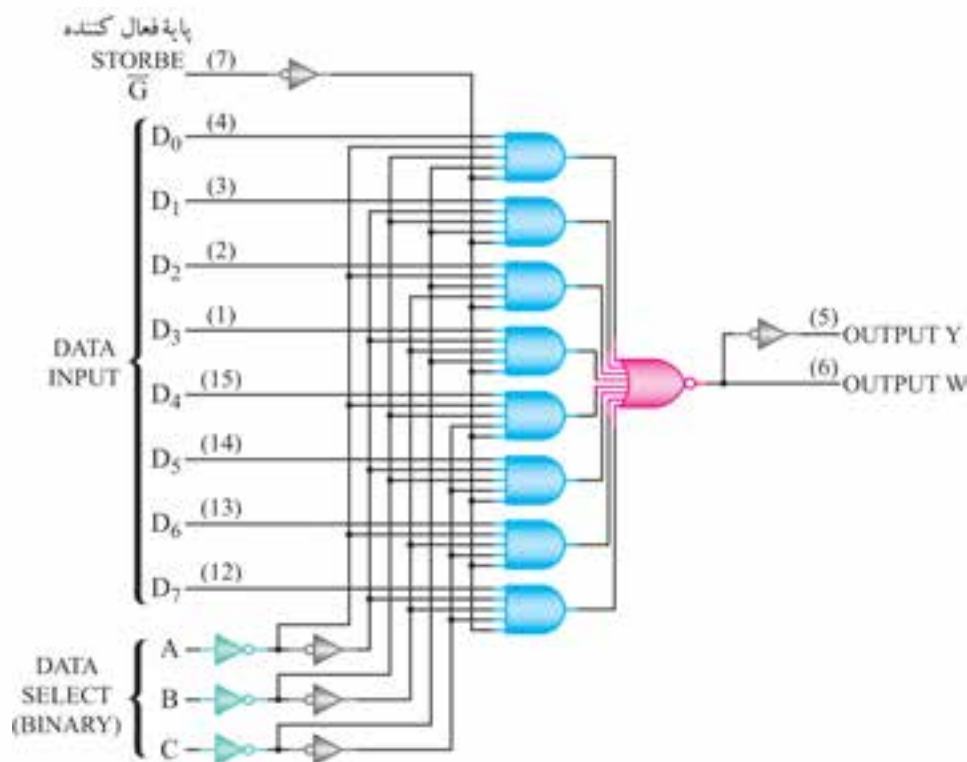
یعنی، F عیناً از D_0 تبعیت می کند، به عبارت دیگر، آدرس $A_1 A_0 = 10$ دروازه شماره (۳) را برای عبور D_0 باز می کند. بقیه حالت های تابع را نیز به همین ترتیب می توانیم، مشخص کنیم. در جدول ۴-۲۱ کلیه حالت های تابع مشخص شده است.

در اغلب مالتی پلکسرها علاوه بر ورودی های آدرس، یک خط کنترل اضافی نیز پیش بینی شده است. این ورودی اضافی با نام های فعال ساز (تواناساز) Enable مشخص می شود.

چنانچه این ورودی در یک حالت از پیش تعریف شده «۰» یا «۱» باشد، همه دروازه های AND را مسدود می سازد و از انتقال اطلاعات ورودی به خروجی مدار پیش گیری می کند. به عبارت دیگر، این ورودی مقدم بر ورودی های آدرس است. در شکل ۴-۴۹ یک مالتی پلکسر ۱ → ۲ با ورودی تواناساز به همراه جدول صحت آن نشان داده شده است.

جدول ۴-۲۲ جدول صحت مالتی پلکسر ۱ → ۲

A_0	EN	f
X	1	0
0	0	D_0
1	0	D_1



پ) مدارهای داخلی

شکل ۴-۵۰ مدار مالتی پلکسر ۱ → ۸ تجارتي (SN ۷۴۱۵۱)

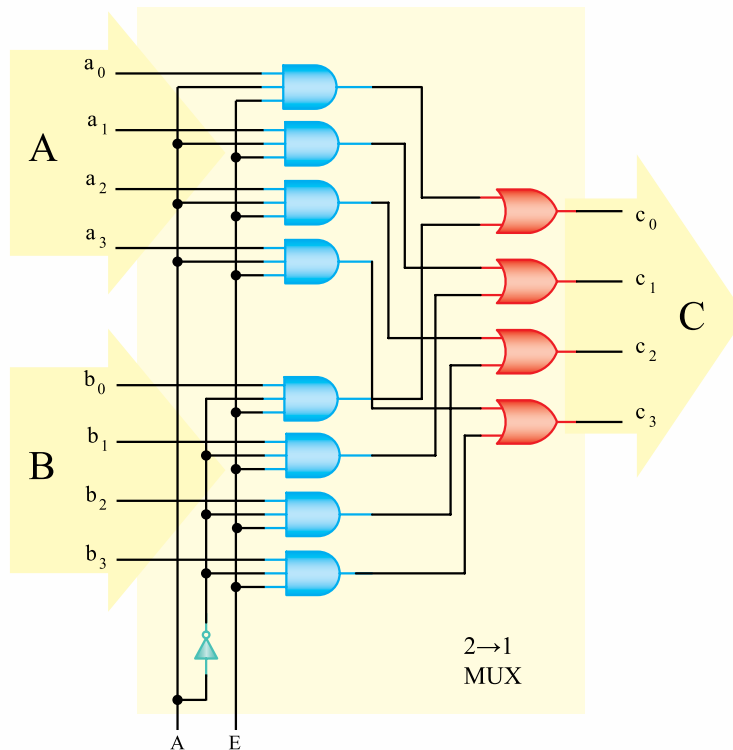
در کامپیوترها مالتی پلکسرها به عنوان گذرگاه داده‌ها Data Bus ایفای نقش می‌کنند. در این موارد، معمولاً به مالتی پلکسرهای چند بیتی نیاز داریم؛ مثلاً، اگر بخواهیم اطلاعات را توسط یک خط انتقال از دو سیستم A و B به سیستم C برسانیم و این اطلاعات در قالب کلمات چهار بیتی باشند، به یک مالتی پلکسر ۱ → ۲، ۴ بیتی نیاز داریم. امکان عرضه مالتی پلکسرهای چند بیتی به صورت تراشه‌های تجارتي وجود ندارد دلیل این امر، افزایش تعداد پایه‌های ضروری آنهاست. لذا باید با استفاده از مالتی پلکسرهای یک بیتی، مالتی پلکسر مورد نیاز خود را طراحی کنیم. در مورد مثال فوق، با استفاده از یک تراشه ۷۴۱۵۷ که شامل چهار مالتی پلکسر ۱ → ۲ یک بیتی است، یک مالتی پلکسر ۱ → ۲ چهار بیتی مطابق شکل ۵۱-۴-الف، ب، پ، ت، و ث طراحی می‌کنیم؛ یعنی کلاً چهار مالتی پلکسر یک بیتی را به صورت موازی به کار می‌بریم.

جدول صحت آی سی SN ۷۴۱۵۱ را در جدول ۲۳-۴ مشاهده می‌کنید.

جدول ۲۳-۴- جدول صحت مالتی پلکسر SN ۷۴۱۵۱

inputs				outputs	
SELECT			STROBE \overline{G}	Y	W
C	B	A			
X	X	X	H	L	\overline{H}
L	L	L	L	D0	$\overline{D0}$
L	L	H	L	D1	$\overline{D1}$
L	H	L	L	D2	$\overline{D2}$
L	H	H	L	D3	$\overline{D3}$
H	L	L	L	D4	$\overline{D4}$
H	L	H	L	D5	$\overline{D5}$
H	H	L	L	D6	$\overline{D6}$
H	H	H	L	D7	$\overline{D7}$

همان‌طور که در جدول ۲۳-۴ مشاهده می‌کنید خروجی W معکوس خروجی Y است.

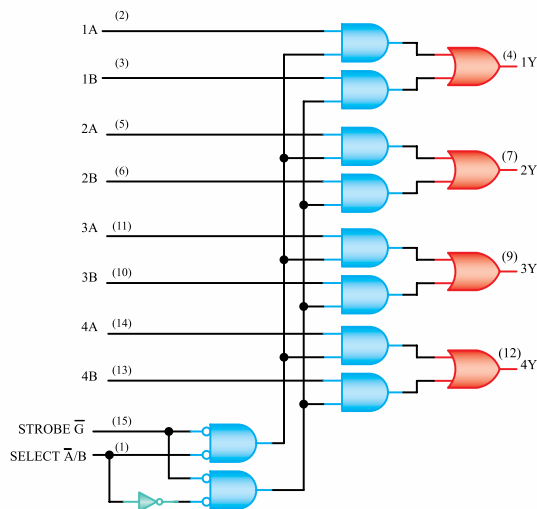


الف) مدار اصلی یک Mux، $2 \rightarrow 1$ چهار بیتی که با استفاده از چهار Mux، $2 \rightarrow 1$ یک بیتی ساخته شده است.

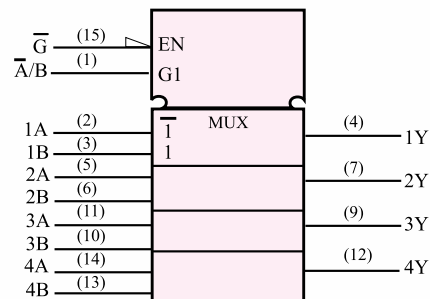
INPUTS				OUTPUT Y
STROBE \overline{G}	SELECT $\overline{A/B}$	A	B	"157, "LS157, "S157
H	X	X	X	L
L	L	L	X	L
L	L	H	X	H
L	H	X	L	L
L	H	X	H	H

H = high level , L = Low level , X = irrelevant

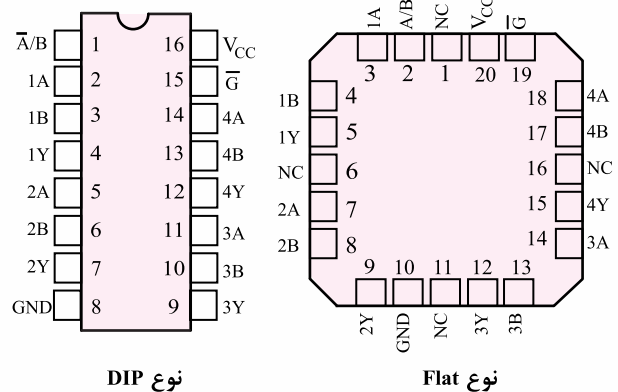
ت) جدول صحت



ث) مدارهای داخلی

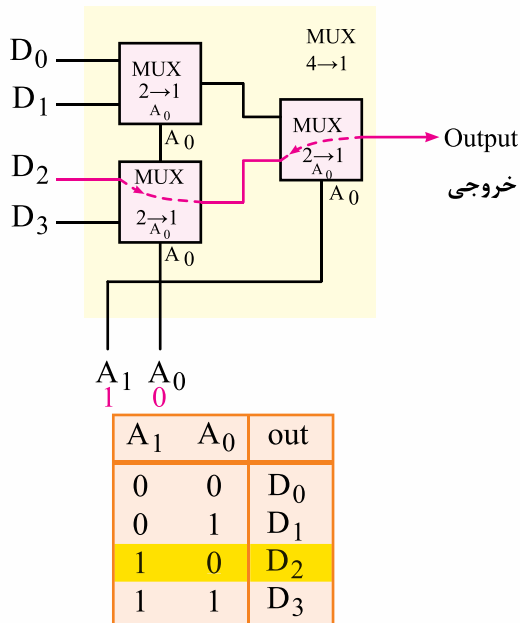


ب) نماد استاندارد IEEE/ANSI



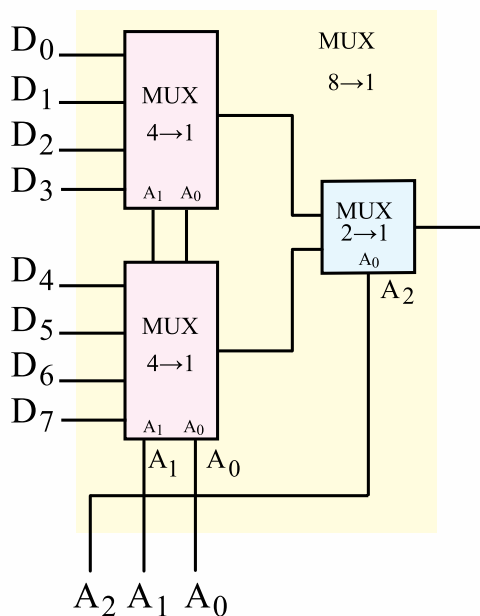
پ) نمای بالا و شماره پایه‌ها

شکل (۵۱-۴) الف چگونگی ساخت Mux چندبیتی؛ ب تا ث- مشخصات تراشه تجاری SNV۴۱۵۷



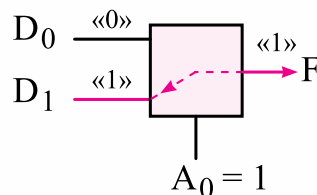
شکل ۴-۵۳- یک مالتی پلکسر ۱ → ۴ همراه جدول صحت آن

مثال ۸-۴: یک مالتی پلکسر ۱ → ۸ را با دو عدد مالتی پلکسر ۱ → ۴ و یک مالتی پلکسر ۱ → ۲ طراحی کنید (شکل ۴-۵۴).



شکل ۴-۵۴- مدار مثال ۸-۴

همان طور که گفته شد، افزایش تعداد پایه های ورودی مالتی پلکسر در طراحی مدارهای منطقی ضروری است. بنابراین در این بخش نحوه توسعه ورودی های یک مالتی پلکسر را با طراحی یک مالتی پلکسر ۱ → ۴ به وسیله سه عدد مالتی پلکسر ۱ → ۲ تشریح می کنیم. در شکل ۴-۵۲ بلوک دیاگرام یک مالتی پلکسر ۱ → ۲ را همراه با جدول صحت آن در جدول ۴-۲۴ مشاهده می کنید. در این مدار به ازای آدرس $A_0 = 1$ ورودی D_1 به خط خروجی وصل شده است.



شکل ۴-۵۲- بلوک دیاگرام مالتی پلکسر ۱ → ۲

جدول ۴-۲۴- جدول صحت مالتی پلکسر ۱ → ۲

A_0	F
0	D_0
1	D_1

اگر سه عدد مالتی پلکسر ۱ → ۲ را مطابق شکل ۴-۵۳ به یکدیگر اتصال دهیم یک مالتی پلکسر ۱ → ۴ شکل می گیرد.

شکل ۴-۵۳، خط آدرس A_0 مالتی پلکسرهای ورودی با یکدیگر موازی شده اند و خط آدرس کم ارزش تر A_1 مالتی پلکسر ۱ → ۴ را تشکیل می دهد و خط آدرس A_2 مالتی پلکسر سوم (خروجی) به عنوان خط آدرس با ارزش تر A_1 مالتی پلکسر ۱ → ۴ قرار می گیرد.

یکی از کاربردهای مالتی پلکسر، اجرای توابع منطقی است. برای اجرای یک تابع منطقی n متغیری می‌توان از یک مالتی پلکسر با n خط آدرس یا $n-1$ خط آدرس یا $n-2$ خط آدرس استفاده کرد. در هر مورد باید تابع را بر حسب n متغیر یا $n-1$ متغیر یا $n-2$ متغیری که به خطوط آدرس مالتی پلکسر وصل می‌شوند، بسط نرمال داد. برای سهولت کار معمولاً متغیرها را به پایه‌های آدرس متناظر آنها وصل می‌کنند. مثلاً اگر بخواهیم تابع $F(A,B,C) = \sum_m(0,2,4,5,7)$ را که در آن ارزش متغیرها به ترتیب $A=4$, $B=2$ و $C=1$ است، به کمک یک مالتی پلکسر $1 \rightarrow 8$ اجرا کنیم باید کم‌ارزش‌ترین متغیر را به کم‌ارزش‌ترین آدرس مالتی پلکسر وصل کنیم یعنی باید C را به پایه A و B را به پایه A_1 و A را به پایه A_2 آدرس مالتی پلکسر اتصال دهیم. (پایه‌های A , A_1 و A_2 پایه‌های آدرس مالتی پلکسر است). با ذکر چند مثال، چگونگی استفاده از مالتی پلکسر را نشان می‌دهیم.

مثال ۱۰-۴:

تابع منطقی $F(A,B,C) = \overline{A}B + BC + A(\overline{C} + B)$ را به کمک یک مالتی پلکسر $1 \rightarrow 8$ اجرا کنید.

حل:

$$F = \overline{A}B + BC + A(\overline{C} + B)$$

$$F = \overline{A}B + BC + A\overline{C} + AB$$

تابع را به صورت استاندارد می‌نویسیم.

$$F = \overline{A}B(\overline{C} + C) + BC(\overline{A} + A) + A\overline{C}(\overline{B} + B) + AB(\overline{C} + C)$$

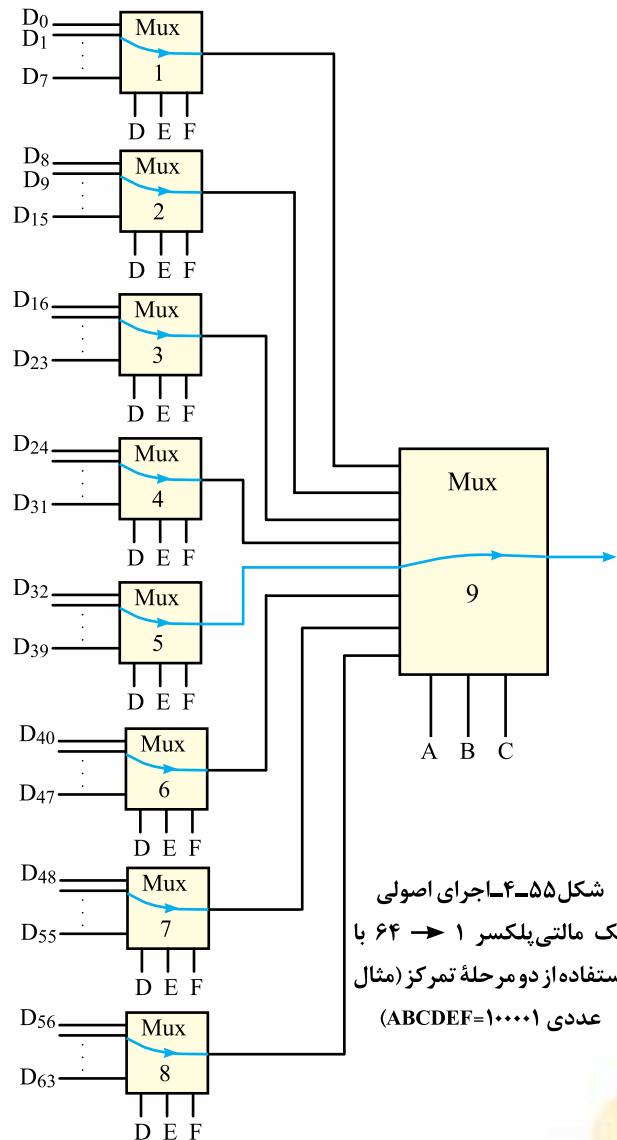
عبارت‌ها را در پرانتز ضرب می‌کنیم.

$$F = \overline{A}B\overline{C} + \overline{A}BC + \overline{A}B\overline{C} + \overline{A}BC + A\overline{C}\overline{B} + A\overline{C}B + ABC + ABC$$

بزرگ‌ترین مالتی پلکسری که در دسترس قرار دارد و به صورت آی‌سی (IC) به بازار عرضه شده است، یک مالتی پلکسر $1 \rightarrow 16$ با شماره تجارتي ۷۴۱۵۰ است. در صورتی که بخواهیم ورودی‌های بیشتری را متمرکز سازیم، باید از چند مالتی پلکسر استفاده کنیم.

مثال ۹-۴: با استفاده از ۹ مالتی پلکسر $1 \rightarrow 8$ یک مالتی پلکسر $1 \rightarrow 64$ بسازید.

حل: ابتدا هر ۸ ورودی را به یک مالتی پلکسر $1 \rightarrow 8$ روی یک خط متمرکز می‌کنیم. سپس ۸ خط متمرکز شده را به کمک یک مالتی پلکسر $1 \rightarrow 8$ دیگر دوباره روی یک خط متمرکز می‌سازیم. در شکل ۴-۵۵ اجرای اصولی چنین مالتی پلکسری نشان داده شده است.



جملات تکراری را حذف می‌کنیم.

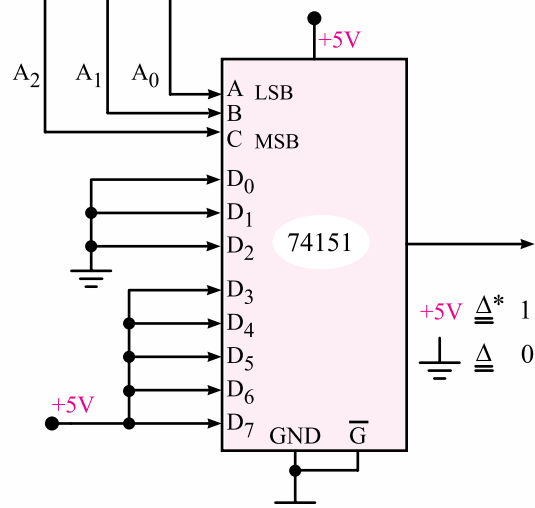
$$F = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

۴ ۵ ۳ ۷ ۶

$$\Sigma m(4, 5, 6, 7)$$

مطابق شکل ۴-۵۶ باید ورودی‌های متناظر با هر یک از جملات تشکیل دهنده تابع «۱» منطقی شود. لذا این ورودی‌ها (ورودی‌های D_7 تا D_4) را به $V_{cc} + 7$ و بقیه ورودی‌ها (ورودی D_3 و D_2 به ترتیب متناظر با جملات $\overline{A}\overline{B}\overline{C}$ ، $\overline{A}\overline{B}C$ و $\overline{A}B\overline{C}$ را به زمین وصل می‌کنیم). توجه کنید که متغیر C در جدول صحت کم‌ارزش‌ترین متغیر تابع است و به همین دلیل، به ورودی آدرس A. یعنی کم‌ارزش‌ترین ورودی آدرس وصل شده است.

A	B	C	آدرس انتخاب شده	خروجی مورد نیاز
0	0	0	D_0	0
0	0	1	D_1	0
0	1	0	D_2	0
0	1	1	D_3	1 $\rightarrow \overline{A}BC$
1	0	0	D_4	1 $\rightarrow \overline{A}\overline{B}C$
1	0	1	D_5	1 $\rightarrow \overline{A}B\overline{C}$
1	1	0	D_6	1 $\rightarrow A\overline{B}\overline{C}$
1	1	1	D_7	1 $\rightarrow ABC$



شکل ۴-۵۶ مدار و جدول مالتی‌پلکسر ۱ \rightarrow ۸ مثال ۴-۱۰

۴-۹- الگوی پرسش

۱- تعداد خط خروجی دودویی یک رمزگذار با ۲۰ حالت را محاسبه کنید.

۲- تعداد خط آدرس دهی یک مالتی‌پلکسر ۵۱۲ \rightarrow ۱ را محاسبه کنید.

۳- یک مالتی‌پلکسر ۱ \rightarrow ۴ را با ورودی تواناساز صفر فعال رسم کنید.

۴- تابع $F(A,B,C) = \Sigma m(0, 1, 4, 6)$ را با یک مالتی‌پلکسر ۱ \rightarrow ۸ اجرا کنید.



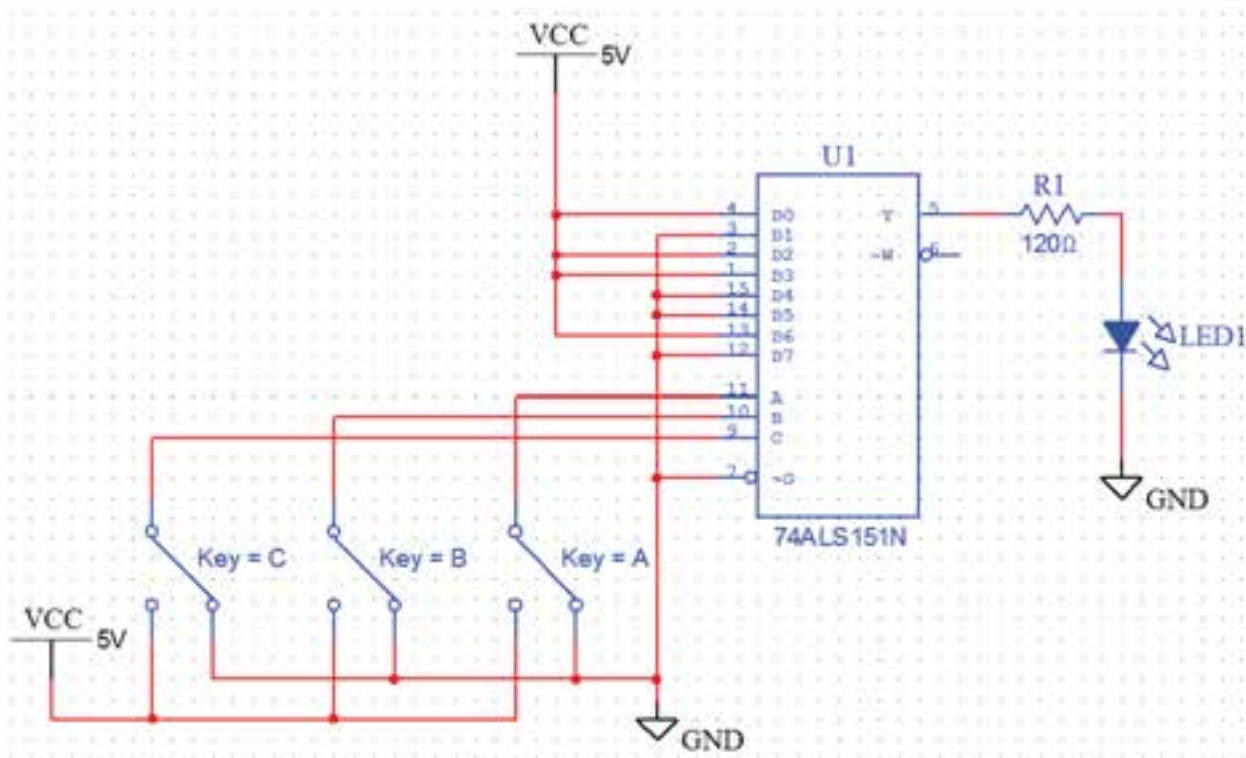
جهت هنجاریان علاقه‌مند: ۵- جدول ۴-۲۵
جدول صحت یک جمع‌کننده کامل است. که S حاصل جمع (Sum) و C رقم نقلی (Carry) جمع‌کننده کامل را مشخص می‌کند. مطابق این جدول مدار خروجی را با استفاده از رمزگشا طراحی کنید.

جدول ۴-۲۵- جدول صحت یک جمع‌کننده کامل

A	B	C_{in}	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

۶- رابطه خروجی S و C را بر اساس جدول ۴-۲۵ بنویسید.

۷- تابع F مدار شکل ۴-۵۷ را که با آی‌سی مالتی‌پلکسر ۷۴۱۵۱ اجرا شده است، بنویسید.



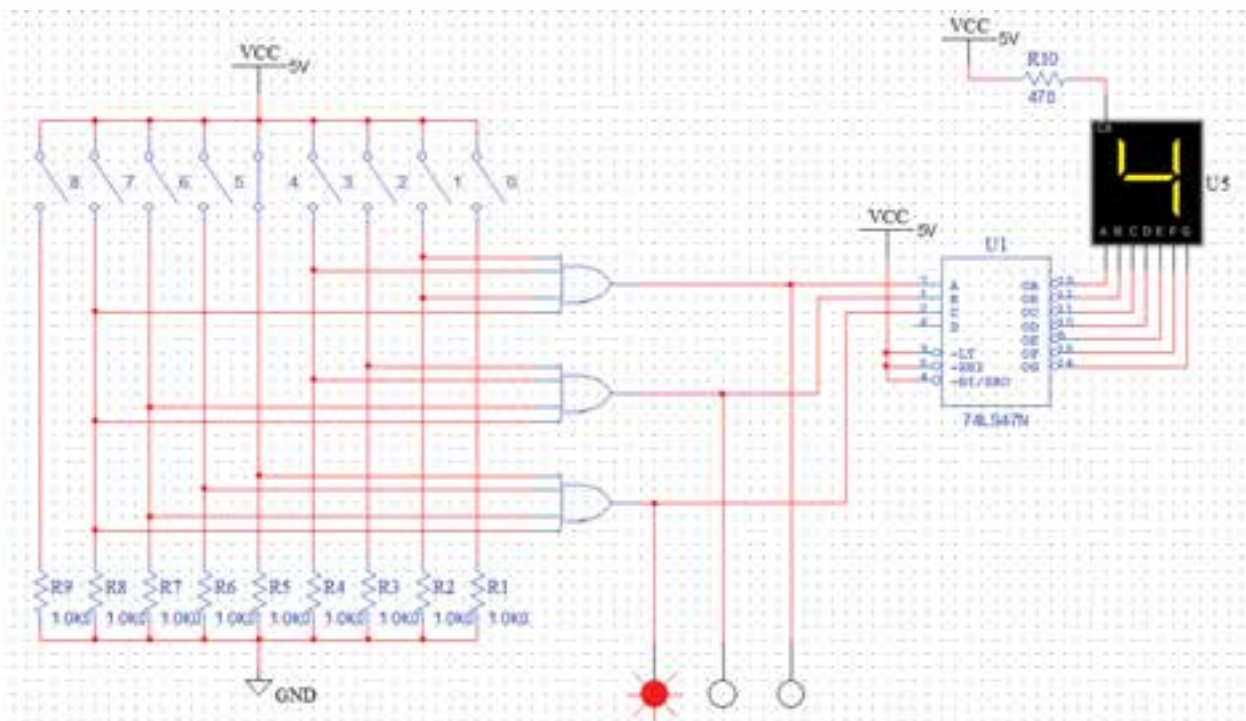
شکل ۴-۵۷- مدار سؤال ۷ الگوی پرسش

صفحه نمایش مشاهده کنید. تمرین ۵ الگوی پرسش را

۴-۱۰- کار با نرم افزار

با نرم افزار اجرا کنید.

مدار شکل ۴-۵۸ رمزگذار است، مدار را توسط نرم افزار مولتی سیم ببندید و صحت عملکرد آن را در



شکل ۴-۵۸- مدار رمزگذار